Timed Basic Parallel Processes

Lorenzo Clemente
University of Warsaw, Poland

Piotr Hofman
University of Warsaw, Poland

Patrick Totzke
University of Liverpool, UK

Abstract

Timed basic parallel processes (TBPP) extend communication-free Petri nets (aka. BPP or commutative context-free grammars) by a global notion of time. TBPP can be seen as an extension of timed automata (TA) with context-free branching rules, and as such may be used to model networks of independent timed automata with process creation.

We show that the coverability and reachability problems (with unary encoded target multiplicities) are PSPACE-complete and EXPTIME-complete, respectively. For the special case of 1-clock TBPP, both are NP-complete and hence not more complex than for untimed BPP. This contrasts with known super-Ackermannian-completeness and undecidability results for general timed Petri nets.

As a result of independent interest, and basis for our NP upper bounds, we show that the reachability relation of 1-clock TA can be expressed by a formula of polynomial size in the existential fragment of linear arithmetic, which improves on recent results from the literature.

2012 ACM Subject Classification Theory of computation → Timed and hybrid models

Keywords and phrases Timed Automata, Petri Nets

Related Version This is the full version of a paper presented at the 30th International Conference on Concurrency Theory (CONCUR 2019).

Funding *Lorenzo Clemente*: Partially supported by Polish NCN grant 2017/26/D/ST6/00201.
Piotr Hofman: Partially supported by Polish NCN grant 2016/21/D/ST6/01368.

Acknowledgements Many thanks to Rasmus Ibsen-Jensen for helpful discussions and pointing us towards [28].

1 Introduction

We study safety properties of unbounded networks of timed processes, where time is global and elapses at the same rate for every process. Each process is a *timed automaton* (TA) [7] controlling its own set of private clocks, not accessible to the other processes. A process can dynamically create new sub-processes, which are thereafter independent from each other and their parent, and can also terminate its execution and disappear from the network.

While such systems can be conveniently modelled in *timed Petri nets* (TdPN), verification problems for this model are either undecidable or prohibitively complex: The reachability problem is undecidable even when individual processes carry only one clock [42] and the coverability problem is undecidable for two or more clocks. In the one-clock case coverability remains decidable but its complexity is hyper-Ackermannian [6, 27].

These hardness results however require unrestricted synchronization between processes, which motivates us to study of the communication-free fragment of TdPN, called *timed basic parallel processes* (TBPP) in this paper. This model subsumes both TA and communication-free Petri nets (a.k.a. BPP [14, 23]). The general picture that we obtain is that extending

communication-free Petri nets by a global notion of time comes at no extra cost in the complexity of safety checking, and it improves on the prohibitive complexities of TdPN.

Our contributions. We show that the TBPP coverability problem is PSPACE-complete, matching same complexity for TA [7, 24], and that the more general TBPP reachability problem is EXPTIME-complete, thus improving on the undecidability of TdPN. The lower bounds already hold for TBPP with two clocks if constants are encoded in binary; EXPTIME-hardness for reachability with no restriction on the number of clocks holds for constants in $\{0, 1\}$. The upper bounds are obtained by reduction to TA reachability and reachability games [31], and assume that process multiplicities in target configurations are given in unary.

In the single-clock case, we show that both TBPP coverability and reachability are NP-complete, matching the same complexity for (untimed) BPP [23]. This paves the way for the automatic verification of unbounded networks of 1-clock timed processes, which is currently lacking in mainstream verification tools such as UPPAAL [35] and KRONOS [47]. The NP lower bound already holds when the target configuration has size 2; when it has size one, 1-clock TBPP coverability becomes NL-complete, again matching the same complexity for 1-clock TA [34] (and we conjecture that 1-clock reachability is in PTIME under the same restriction).

As a contribution of independent interest, we show that the ternary reachability relation of 1-clock TA can be expressed by a formula of existential linear arithmetic (∃LA) of polynomial size. By *ternary reachability relation* we mean the family of relations $\{\rightarrow_{pq}\}$ s.t. $\mu \xrightarrow{\delta}_{pq} \nu$ holds if from control location p and clock valuation $\mu \in \mathbb{R}^k_{\geq 0}$ it is possible to reach control location q and clock valuation $\nu \in \mathbb{R}^k_{\geq 0}$ in exactly $\delta \in \mathbb{R}_{\geq 0}$ time. This should be contrasted with analogous results (cf. [26]) which construct formulas of exponential size, even in the case of 1-clock TA. Since the satisfiability problem for ∃LA is decidable in NP, we obtain a NP upper bound to decide ternary reachability $\rightarrow_{pq}$. We show that the logical approach is optimal by providing a matching NP lower bound for the same problem. Our NP upper bounds for the 1-clock TBPP coverability and reachability problems are obtained as an application of our logical expressibility result above, and the fact that ∃LA is in NP; as a further technical ingredient we use polynomial bounds on the piecewise-linear description of value functions in 1-clock priced timed games [28].

Related research. Starting from the seminal PSPACE-completeness result of the nonemptiness problem for TA [7] (cf. also [24]), a rich literature has emerged considered more challenging verification problems, including the symbolic description of the reachability relation [19, 21, 32, 22, 26]. There are many natural generalizations of TA to add extra modelling capabilities, including *time Petri Nets* [37, 39] (which associate timing constraints to transitions) the already mentioned timed Petri nets (TdPN) [42, 6, 27] (where tokens carry clocks which are tested by transitions), *networks of timed processes* [5], several variants of *timed pushdown automata* [12, 20, 8, 44, 4, 41, 9, 18, 17], *timed communicating automata* [33, 16, 3, 15], and their lossy variant [1], and timed process calculi based on Milners CCS (e.g. [10]). While decision problems for TdPN have prohibitive complexity/are undecidable, it has recently been shown that structural safety properties are PSPACE-complete using forward accelerations [2].

Outline. In Section 2 we define TBPP and their reachability and coverability decision problems. In Section 3 we show that the reachability relation for 1-clock timed automata can be expressed in polynomial time in an existential formula of linear arithmetic, and that the latter logic is in NP. We apply this result in Sec. 4 to show that the reachability and

coverability problems for 1-clock TBPP is NP-complete. Finally, in Section 5 we study the case of TBPP with $k \geq 2$ clocks, and in Section 6 we draw conclusions.

2 Preliminaries

Notations. We use $\mathbb{N}$ and $\mathbb{R}_{\geq 0}$ to denote the sets of nonnegative integers and reals, respectively. For $c \in \mathbb{R}_{\geq 0}$ we write $int(c) \in \mathbb{N}$ for its integer part and $frac(c) \stackrel{def}{=} c - int(c)$ for its fractional part. For a set $\mathcal{X}$, we use $\mathcal{X}^*$ to denote the set of finite sequences over $\mathcal{X}$ and $\mathcal{X}^\oplus$ to denote the set of finite multisets over $\mathcal{X}$, i.e., functions $\mathbb{N}^{\mathcal{X}}$. We denote the empty multiset by $\emptyset$, we denote the union of two multisets $\alpha, \beta \in \mathbb{N}^{\mathcal{X}}$ by $\alpha + \beta$, which is defined point-wise, and by $\alpha \leq \beta$ we denote the natural partial order on multisets, also defined point-wise. The *size* of a multiset $\alpha \in \mathbb{N}^{\mathcal{X}}$ is $|\alpha| = \sum_{X \in \mathcal{X}} \alpha(X)$. We overload notation and we let $X \in \mathcal{X}$ denote the singleton multiset of size 1 containing element X. For example, if $X, Y \in \mathcal{X}$, then the multiset consisting of 1 occurrence of X and 2 of Y will be denoted by $\alpha = X + Y + Y$; it has size $|\alpha| = 3$.

Clocks. Let $\mathcal{C}$ be a finite set of *clocks*. A *clock valuation* is a function $\mu \in \mathbb{R}_{\geq 0}^{\mathcal{C}}$ assigning a nonnegative real to every clock. For $t \in \mathbb{R}_{\geq 0}$, we write $\mu + t$ for the valuation that maps clock $x \in \mathcal{C}$ to $\mu(x) + t$. For a clock $x \in \mathcal{C}$ and a clock or constant $e \in \mathcal{C} \cup \mathbb{N}$ let $\mu[x := e]$ be the valuation ν s.t. $\nu(x) = \mu(e)$ and $\nu(z) = \mu(z)$ for every other clock $z \neq x$ (where we assume $\mu(k) = k$ for a constant $k \in \mathbb{N}$); for a sequence of assignments $R = (x_1 := e_1; \cdots; x_n := e_n)$ let $\mu[R] = \mu[x_1 := e_1] \cdots [x_n := e_n]$. A *clock constraint* is a conjunction of linear inequalities of the form $c \bowtie k$, where $c \in \mathcal{C}$, $k \in \mathbb{N}$, and $\bowtie \in \{<, \leq, =, \geq, >\}$; we also allow **true** for the trivial constraint which is always satisfied. We write $\mu \models \varphi$ to denote that the valuation μ satisfies the constraint φ.

Timed basic parallel processes. A *timed basic parallel process* (TBPP) consists of finite sets $\mathcal{C}$, $\mathcal{X}$, and $\mathcal{R}$ of clocks, nonterminal symbols, and rules. Each rule is of the form

$$X \stackrel{\varphi;R}{\Longrightarrow} \alpha$$

where $X \in \mathcal{X}$ is a nonterminal, φ is a clock constraint, R is a sequence of assignments of the form $x := e$, where e is either a constant in $\mathbb{N}$ or a clock in $\mathcal{C}$, and $\alpha \in \mathcal{X}^\oplus$ is a finite multiset of successor nonterminals[1]. Whenever the test $\varphi \equiv$ **true** is trivial, or R is the empty sequence, we just omit the corresponding component and just write $X \stackrel{\varphi}{\Longrightarrow} \alpha$, $X \stackrel{R}{\Longrightarrow} \alpha$, or $X \Longrightarrow \alpha$. Finally, we say that we *reset* the clock x_i if we assign it to 0.

Henceforth, we assume w.l.o.g. that the size $|\alpha|$ is at most 2. A rule with $\alpha = \emptyset$ is called a *vanishing* rule, and a rule with $|\alpha| = 2$ is called a *branching* rule. We will write k-TBPP to denote the class of TBPP with k clocks.

A *process* is a pair $(X, \mu) \in \mathcal{X} \times \mathbb{R}_{\geq 0}^{\mathcal{C}}$ comprised of a nonterminal X and a clock valuation μ, and a *configuration* α is a multiset of processes, i.e., $\alpha \in (\mathcal{X} \times \mathbb{R}_{\geq 0}^{\mathcal{C}})^\oplus$. For a process $P = (X, \mu)$ and $t \in \mathbb{R}_{\geq 0}$, we denote by $P + t$ the process $(X, \mu + t)$, and for a configuration $\alpha = P_1 + \cdots + P_n$, we denote by $\alpha + t$ the configuration $Q_1 + \cdots + Q_n$, where

[1] We note that clock updates $x := k$ with $k \in \mathbb{N}$ can be encoded with only a polynomial blow-up by replacing them with $x := 0$, while recording in the finite control the last update k, and replacing a test $x \bowtie h$ with $x \bowtie h - k$. We use them as a syntactic sugar to simplify the presentation of some constructions.

$Q_1 = P_1 + t, \ldots, Q_n = P_n + t$. The semantics of a TBPP $(\mathcal{C}, \mathcal{X}, \mathcal{R})$ is given by an infinite timed transition system $(C, \to)$, where $C = (\mathcal{X} \times \mathbb{R}_{\geq 0}^{\mathcal{C}})^{\oplus}$ is the set of configurations, and $\to \subseteq C \times \mathbb{R}_{\geq 0} \times C$ is the transition relation between configurations. There are two kinds of transitions:

Time elapse: For every configuration $\alpha \in C$ and $t \in \mathbb{R}_{\geq 0}$, there is a transition $\alpha \xrightarrow{t} \alpha + t$ in which all clocks in all processes are simultaneously increased by t. In particular, the empty configuration stutters: $\emptyset \xrightarrow{t} \emptyset$, for every $t \in \mathbb{R}_{\geq 0}$.

Discrete transitions: For every configuration $\gamma = \alpha + (X, \mu) + \beta \in C$ and rule $X \overset{\varphi;R}{\Longrightarrow} Y + Z$ s.t. $\mu \models \varphi$ there is a transition $\gamma \xrightarrow{0} \alpha + (Y, \nu) + (Z, \nu) + \beta$, where $\nu = \mu[R]$. Analogously, rules $X \overset{\varphi;R}{\Longrightarrow} Y$ and $X \overset{\varphi;R}{\Longrightarrow} \emptyset$ induce transitions $\gamma \xrightarrow{0} \alpha + (Y, \nu) + \beta$ and $\gamma \xrightarrow{0} \alpha + \beta$.

A *run* starting in α and ending in β is a sequence of transitions $\alpha = \alpha_0 \xrightarrow{t_1} \alpha_1 \cdots \xrightarrow{t_n} \alpha_n = \beta$. We write $\alpha \xrightarrow{t} \beta$ whenever there is a run as above where the sum of delays is $t = t_1 + \cdots + t_n$, and we write $\alpha \xrightarrow{*} \beta$ whenever $\alpha \xrightarrow{t} \beta$ for some $t \in \mathbb{R}_{\geq 0}$.

TBPP generalise several known models: A *timed automaton* (TA) [7] is a TBPP without branching rules; in the context of TA, we will sometimes call nonterminals with the more standard name of *control locations*. Untimed *basic parallel processes* (BPP) [14, 23] are TBPP over the empty set of clocks $\mathcal{X} = \emptyset$. TBPP can also be seen as a structural restriction of *timed Petri nets* [6, 27] where each transition consumes only one token at a time.

TBPP are related to *alternating timed automata* (ATA) [38, 36]: Branching in TBPP rules corresponds to universal transitions in ATA. However, ATA offer additional means of synchronisation between the different branches of a run tree: While in a TBPP synchronisation is possible only through the elapse of time, in an ATA all branches must read the same timed input word.

Decision problems. We are interested in checking safety properties of TBPP in the form of the following decision problems. The *reachability problem* asks whether a target configuration is reachable from a source configuration.

Input: A TBPP $(\mathcal{C}, \mathcal{X}, \mathcal{R})$, an initial $X \in \mathcal{X}$ and target nonterminals $T_1, \ldots, T_n \in \mathcal{X}$.
Question: Does $(X, \vec{0}) \xrightarrow{*} (T_1, \vec{0}) + \cdots + (T_n, \vec{0})$ hold?

It is crucial that we reach all processes in the target configurations *at the same time*, which provides an external form of global synchronisation between processes.

Motivated both by complexity considerations and applications for safety checking, we study the *coverability problem*, where it suffices to reach some configuration larger than the given target in the multiset order. For configurations $\alpha, \beta \in (\mathcal{X} \times \mathbb{R}_{\geq 0}^{\mathcal{C}})^{\oplus}$, let $\alpha \xrightarrow{*} \cdot \geq \beta$ whenever there exists $\gamma \in (\mathcal{X} \times \mathbb{R}_{\geq 0}^{\mathcal{C}})^{\oplus}$ s.t. $\alpha \xrightarrow{*} \gamma \geq \beta$.

Input: A TBPP $(\mathcal{C}, \mathcal{X}, \mathcal{R})$, an initial $X \in \mathcal{X}$ and target nonterminals $T_1, \ldots, T_n \in \mathcal{X}$.
Question: Does $(X, \vec{0}) \xrightarrow{*} \cdot \geq (T_1, \vec{0}) + \cdots + (T_n, \vec{0})$?

The *simple* reachability/coverability problems are as above but with the restriction that the target configuration is of size 1, i.e., a single process. Notice that this is a proper restriction, since reachability and coverability do not reduce in general to their simple variant. Finally, the *non-emptiness problem* is the special case of the reachability problem where the target configuration α is the empty multiset $\emptyset$.

In all decision problems above the restriction to zero-valued clocks in the initial process is mere convenience, since we could introduce a new initial nonterminal Y and a transition $Y \xrightarrow{x_1:=\mu(x_1);\ldots;x_n:=\mu(x_n)} X$ initialising the clocks to the initial values provided by the (rational) clock valuation $\mu \in \mathbb{Q}^{\mathcal{C}}_{\geq 0}$. Similarly, if we wanted to reach the final configuration $\alpha = (X_1, \mu_1) + (X_2, \mu_2)$ with $\mu_1, \mu_2 \in \mathbb{Q}^{\mathcal{C}}_{\geq 0}$, then we could add two nonterminals Y_1, Y_2 and two new rules $X_1 \xrightarrow{x_1=\mu_1(x_1)\wedge\cdots\wedge x_n=\mu_1(x_n);x_1:=0;\ldots;x_n:=0} Y_1$ and $X_2 \xrightarrow{x_1=\mu_2(x_1)\wedge\cdots\wedge x_n=\mu_2(x_n);x_1:=0;\ldots;x_n:=0} Y_2$ and check whether $X \xrightarrow{*} (Y_1, \vec{0}) + (Y_2, \vec{0})$ holds. (It is standard to transform TBPP with constraints of the form $x_i = k$ with $k \in \mathbb{Q}_{\geq 0}$ in the form $x_i = k$ with $k \in \mathbb{N}$.) Similarly, the restriction of having just one initial nonterminal process is also w.l.o.g., since if we wanted to check reachability from $(X_1, \vec{0}) + (X_2, \vec{0})$ we could just add a new initial nonterminal X and a branching rule $X \Longrightarrow X_1 + X_2$.

For complexity considerations we will assume that all constants appearing in clock constraints are given in binary encoding, and that the multiplicities of target processes are in unary.

3 Reachability Relations of One-Clock Timed Automata

In this section we show that the reachability relation of 1-clock TA is expressible as an existential formula of linear arithmetic of polynomial size. Since the latter fragment is in NP, this gives an NP algorithm to check whether a family of TA can reach the respective final locations *at the same time*. This result will be applied in Sec. 4 to show that coverability and reachability of 1-TBPP are in NP. We first show that existential linear arithmetic is in NP (which is an observation of independent interest), and then how to express the reachability relation of 1-TA in existential linear arithmetic in polynomial time.

The set of *terms* t is generated by the following abstract grammar

$$s, t \ ::= \ x \mid k \mid \lfloor t \rfloor \mid \mathit{frac}(t) \mid -t \mid s + t \mid k \cdot t,$$

where x is a rational variable, $k \in \mathbb{Z}$ is an integer constant encoded in binary, $\lfloor t \rfloor$ represents the integral part of t, and $\mathit{frac}(t)$ its fractional part. *Linear arithmetic* (LA) is the first order language with atomic proposition of the form $s \leq t$ [46], we denote by $\exists$LA its existential fragment, and by qf-LA its quantifier-free fragment. Linear arithmetic generalises both *Presburger arithmetic* (PA) and *rational arithmetic* (RA), whose existential fragments are known to be in NP [40, 25]. This can be generalised to $\exists$LA. (The same result can be derived from the analysis of [11, Theorem 3.1]).

▶ **Theorem 1.** *The existential fragment $\exists$LA of LA is in* NP.

Let $\mathcal{A} = (\mathcal{C}, \mathcal{X}, \mathcal{R})$ be a k-TA. The *ternary reachability relation* of $\mathcal{A}$ is the family of relations $\{\rightarrow_{XY}\}_{X,Y\in\mathcal{X}}$, where each $\rightarrow_{XY} \subseteq \mathbb{R}^{\mathcal{C}}_{\geq 0} \times \mathbb{R}_{\geq 0} \times \mathbb{R}^{\mathcal{C}}_{\geq 0}$ is defined as: $\mu \xrightarrow{\delta}_{XY} \nu$ iff $(X, \mu) \xrightarrow{\delta} (Y, \nu)$. We say that the reachability relation is *expressed* by a family of LA formulas $\{\varphi_{XY}\}_{X,Y\in\mathcal{X}}$ if

$$\mu \xrightarrow{\delta}_{XY} \nu \quad \text{iff} \quad (\mu, \delta, \nu) \models \varphi_{XY}(\vec{x}, t, \vec{y}), \text{ for every } X, Y \in \mathcal{X}, \mu, \nu \in \mathbb{R}^{\mathcal{C}}_{\geq 0}, \delta \in \mathbb{R}_{\geq 0}.$$

In the formula $\varphi_{XY}(\vec{x}, t, \vec{y})$, $\vec{x}$ are k variables representing the clock values in location X at the beginning of the run, $\vec{y}$ are k variables representing the clock values in location Y at the end of the run, and t is a single variable representing the total time elapsed during the run. In the rest of this section, we assume that the TA has only one clock $\mathcal{X} = \{x\}$.

The main result of this section is that 1-TA reachability relations are expressible by ∃LA formulas constructible in polynomial time.

▶ **Theorem 2.** *Let $\mathcal{A}$ be a 1-TA. The reachability relation $\{\rightarrow_{XY}\}_{X,Y\in\mathcal{X}}$ is expressible as a family of formulas $\{\varphi_{XY}\}_{X,Y\in\mathcal{X}}$ of existential linear arithmetic ∃LA in polynomial time.*

In the rest of the section we prove the theorem above. We begin with some preliminaries.

Interval abstraction. We replace the integer value of the clock x by its interval [34]. Let $0 = k_0 < k_1 < \cdots < k_n < k_{n+1} = \infty$ be all integer constants appearing in constraints of $\mathcal{A}$, and let the set of intervals be the following totally ordered set:

$$\Lambda = \{\{k_0\} < (k_0, k_1) < \{k_1\} < \cdots < (k_{n-1}, k_n) < \{k_n\} < (k_n, k_{n+1})\}\,.$$

Clearly, we can resolve any constraint of $\mathcal{A}$ by looking at the interval $\lambda \in \Lambda$. We write $\lambda \models \varphi$ whenever $v \models \varphi$ for some $v \in \lambda$ (whose choice does not matter by the definition of λ).

The construction. Let $\mathcal{A} = (\{x\}, \mathcal{X}, \mathcal{R})$ be a TA. In order to simplify the presentation below, we assume w.l.o.g. that the only clock updates are resets $x := 0$ (cf. footnote 1). We build an NFA $\mathcal{B} = (\Sigma, Q, \rightarrow)$ where Σ contains symbols (r, ε) and $(r, \checkmark_\lambda)$ for every transition $r \in \mathcal{R}$ of $\mathcal{A}$ and interval $\lambda \in \Lambda$, and an additional symbol τ representing time elapse, and $Q = \mathcal{X} \times \Lambda$ is a set of states of the form (X, λ), where $X \in \mathcal{X}$ is a control location of $\mathcal{A}$ and $\lambda \in \Lambda$ is an interval. Transitions $\rightarrow \subseteq Q \times \Sigma \times Q$ are defined as follows. A rule $r = X \xrightarrow{\varphi;R} Y \in \mathcal{R}$ of $\mathcal{A}$ generates one or more transitions in $\mathcal{B}$ of the form

$$(X, \lambda) \xrightarrow{(r,a)} (Y, \mu)$$

whenever $\lambda \models \varphi$ and any of the following two conditions is satisfied:

- the clock is not reset i.e R is equal $x := x$, and $\mu = \lambda, a = \varepsilon$, or
- the clock is reset $x := 0$, $\mu = \{0\}$, and the automaton emits a tick $a = \checkmark_\lambda$.

A time elapse transition is simulated in $\mathcal{B}$ by transitions of the form

$$(X, \lambda) \xrightarrow{\tau} (X, \mu), \qquad \lambda \leq \mu \text{ (the total ordering on intervals).}$$

Reachability relation of $\mathcal{A}$. For a set of finite words $L \subseteq \Sigma^*$, let $\psi_L(\vec{y})$ be a formula of existential Presburger arithmetic with a free integral variable y_λ for every interval $\lambda \in \Lambda$ counting the number of symbols of the form $(r, \checkmark_\lambda)$, for some $r \in \mathcal{R}$. The formula ψ_L can be computed from the Parikh image of L: By [45, Theorem 4], a formula $\tilde{\psi}_L(\vec{z})$ of existential Presburger arithmetic can be computed in linear time from an NFA (or even a context-free grammar) recognising L, and then one just defines $\psi_L(\vec{y}) \equiv \exists \vec{z} \cdot \tilde{\psi}_L(\vec{z}) \wedge \bigwedge_{\lambda\in\Lambda} y_\lambda = \sum_{r\in\mathcal{R}} z_{r,\lambda}$. Let L_{cd} be the regular language recognised by $\mathcal{B}$ by making c initial and d final, and let $\Lambda = \{\lambda_0, \ldots, \lambda_{2n+1}\}$ contain $2n+2$ intervals. Let $\psi_{cd}(x, t, x')$ be a formula of existential Presburger arithmetic computing the total elapsed time t, given the initial x and final x' values of the unique clock:

$$\begin{aligned}
\psi_{cd}(x,t,x') \;\equiv\; & \exists y_0, \ldots, y_{2n+1} \cdot \psi_{L_{cd}}(\lfloor y_0 \rfloor, \ldots, \lfloor y_{2n+1} \rfloor) \;\wedge \\
& \exists z_0, \ldots, z_{2n+1} \cdot \bigwedge_{\lambda_i \in \Lambda} (z_i \in y_i \cdot \lambda_i) \;\wedge\; t = x' - x + \sum_{\lambda_i \in \Lambda} z_i, \text{ where} \\
z \in y \cdot \lambda \;\equiv\; & \begin{cases} a \cdot y < z < b \cdot y & \text{if } \lambda = (a, b), \\ z = a \cdot y & \text{if } \lambda = \{a\}\,. \end{cases}
\end{aligned}$$

Intuitively, y_i represents the total number of times the clock is reset while in interval λ_i, and z_i represents the sum of the values of the clock when it is reset in interval λ_i. For control locations X, Y of $\mathcal{A}$, let

$$\varphi_{XY}(x,t,x') \equiv \bigvee_{\lambda,\mu\in\Lambda} \{x \in \lambda \wedge x' \in \mu \wedge \psi_{cd}(x,t,x') \mid c = (X,\lambda), d = (Y,\mu)\}.$$

The correctness of the construction is stated below.

▶ **Lemma 3.** *For every configurations (X,u) and (Y,v) of $\mathcal{A}$ and total time elapse $\delta \geq 0$,*

$$u \xrightarrow{\delta}_{XY} v \quad \textit{iff} \quad (u,\delta,v) \models \varphi_{XY}(x,t,x').$$

We conclude this section by applying Theorem 2 to solve the 1-TA ternary reachability problem. The *ternary reachability problem* takes as input a TA $\mathcal{A}$ as above, with two distinguished control locations $X, Y \in \mathcal{X}$, and a total duration $\delta \in \mathbb{Q}$ (encoded in binary), and asks whether $(\vec{0}) \xrightarrow{\delta}_{XY} (\vec{0})$. The result below shows that computing 1-TA reachability relations is optimal in order to solve the ternary reachability problem.

▶ **Theorem 4.** *The ternary reachability problem for 1-TA is* NP*-complete.*

Proof. For the upper bound, apply Theorem 2 to construct in polynomial time a formula of $\exists$LA expressing the reachability relation and check satisfiability in NP thanks to Theorem 1.

The lower bound can be seen by reduction from SubsetSum. Let $\mathcal{S} = \{a_1, \ldots, a_k\} \subseteq \mathbb{N}$ and $a \in \mathbb{N}$ be the input to the subset sum problem, whereby we look for a subset $\mathcal{S}' \subseteq \mathcal{S}$ s.t. $a = \sum_{b\in\mathcal{S}'} b$. We construct a TA with a single clock x and locations $\mathcal{X} = \{X_0, \ldots, X_k\}$, where X_0 is the initial location and X_k the target. A path through the system describes a subset by spending exactly 0 or a_i time in location X_i (see Figure 1). In the constructed automaton, $(X_1, 0) \xrightarrow{a} (X_k, 0)$ iff the subset sum instance was positive. ◀

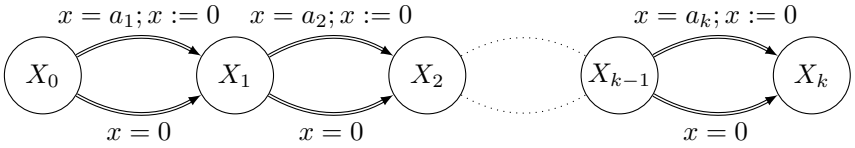

Figure 1 Reduction from subset sum to 1-TA (ternary) reachability. We have $(X_0, 0) \xrightarrow{t} (X_k, 0)$ iff $t = \sum_{b\in\mathcal{S}'} b$ for some subset $\mathcal{S}' \subseteq \{a_1, \ldots, a_k\}$.

4 One-Clock TBPP

As a warm-up we note that the simple coverability problem for 1-TBPP, where the target has size one, is inter-reducible with the reachability problem for 1-clock TA and hence NL-complete [34].

▶ **Theorem 5.** *The simple coverability problem for 1-clock TBPP is* NL*-complete.*

Proof. The lower bound is trivial since 1-TBPP generalize 1-TA. For the other direction we can transform a given TBPP into a TA by replacing branching rules of the form $X \xRightarrow{\varphi,R} Y + Z$ with two rules $X \xRightarrow{\varphi,R} Y$ and $X \xRightarrow{\varphi,R} Z$. In the constructed TA we have $(X,\mu) \xrightarrow{*} (Y,\nu)$ if, and only if, $(X,\mu) \xrightarrow{*} (Y,\nu) + \gamma$ for some γ in the original TBPP. ◀

The construction above works because the target is a single process and so there are no constraints on the other processes in γ, which were produced as side-effects by the branching rules. The (non-simple) 1-TBPP coverability problem is in fact NP-complete. Indeed, even for untimed BPP, coverability is NP-hard, and this holds already when target sets are encoded in unary (which is the setting we are considering here) [23]. We show that for 1-TBPP this lower bound already holds if the target has fixed size 2.

▶ **Lemma 6.** *Coverability is* NP*-hard for 1-TBPP already for target sets of size* ≥ 2*.*

Proof. We proceed by reduction from subset sum as in Theorem 4. The only difference will be that here, we use one extra process to keep track of the total elapsed time. Let $\mathcal{S} = \{a_1, \ldots, a_k\} \subseteq \mathbb{N}$ and $t \in \mathbb{N}$ be the input to the subset sum problem. We construct a 1-TBPP with nonterminals $\mathcal{X} = \{S, X_0, \ldots, X_k, Y\}$, where S is the initial nonterminal and Y will be used to keep track of the total time elapsed. The rules are as in the proof of Theorem 4, and additionally we have an initial branching rule $S \overset{x=0}{\Longrightarrow} X_0 + Y$. We have $(S,0) \xrightarrow{*} \cdot \geq (X_k, 0) + (Y, t)$ if, and only if, $t = \sum_{b \in \mathbb{B}} b$ for some subset $\mathcal{S}' \subseteq \mathcal{S}$. ◀

In the remainder of this section we will argue (Theorem 11) that a matching NP upper bound even holds for the *reachability* problem for 1-TBPP. Let us first motivate the key idea behind the construction. Consider the following TBPP coverability query:

$$(S,0) \xrightarrow{*} \cdot \geq (A,0) + (B,0). \tag{$\dagger$}$$

If ($\dagger$) holds, then there is a derivation tree witnessing that $(S,0) \xrightarrow{*} (A,0) + (B,0) + \gamma$ for some configuration γ. The least common ancestor of leaves $(A,0)$ and $(B,0)$ is some process $(C,c) \in (\mathcal{X} \times \mathbb{R}_{\geq 0})$. Consider the TA $\mathcal{A}$ obtained from the TBPP by replacing branching rules $X \overset{\varphi;R}{\Longrightarrow} X_i + X_j$ with linear rules $X \overset{\varphi;R}{\Longrightarrow} X_i$ and $X \overset{\varphi;R}{\Longrightarrow} X_j$, and let the reachability relation of $\mathcal{A}$ be expressed by ∃LA formulas $\{\varphi_{XY}\}_{X,Y \in \mathcal{X}}$, which are of polynomial size by Theorem 2. Then our original coverability query ($\dagger$) is equivalent to satisfiability the following ∃LA formula:

$$\psi \;\equiv\; \exists t_0, t_1, c \in \mathbb{R} \cdot (\varphi_{SC}(0, t_0, c) \;\wedge \varphi_{CA}(c, t_1, 0) \wedge \varphi_{CB}(c, t_1, 0))\,.$$

More generally, for any coverability query $(S,0) \xrightarrow{*} \cdot \geq \alpha$ the number of common ancestors is linear in $|\alpha|$, and thus we obtain a ∃LA formula ψ of polynomial size, whose satisfiability we can check in NP thanks to Theorem 1.

▶ **Theorem 7.** *The coverability problem for 1-clock TBPP is* NP*-complete.*

In order to witness reachability instances we need to refine the argument above to restrict the TA in such a way that they do not accidentally produce processes that cannot be removed in time. To illustrate this point, consider a 1-TBPP with rules

$$X \overset{x=0}{\Longrightarrow} Y + Z \qquad \text{and} \qquad Z \overset{x>0}{\Longrightarrow} \emptyset.$$

Clearly $(X,0) \xrightarrow{*} (Y,0)$ holds in the TA with rules $X \overset{x=0}{\Longrightarrow} Y$ and $X \overset{x=0}{\Longrightarrow} Z$ instead of the branching rule above. In the TBPP however, $(X,0)$ cannot reach $(Y,0)$ because the branching rule produces a process $(Z,0)$, which needs a positive amount of time to be rewritten to $\emptyset$.

▶ **Definition 8.** *For a nonterminal* X *let* $\mathit{Vanish}_X \subseteq \mathbb{R}^2$ *be the binary predicate such that*

$$\mathit{Vanish}_X(x,t) \quad \textit{if} \quad (X,x) \xrightarrow{t} \emptyset$$

Intuitively, $\mathit{Vanish}_X(x,t)$ holds if the configuration (X,x) can vanish in time at most t.

The time it takes to remove a processes (Z,z) can be computed as the value of a one-clock priced timed game [13, 28]. These are two-player games played on 1-clock TA where players aim to minimize/maximize the cost of a play leading up to a designated target state. Nonnegative costs may be incurred either by taking transitions, or by letting time elapse. In the latter case, the incurred cost is a linear function of time, determined by the current control-state. Bouyer et al. [13] prove that such games admit ε-optimal strategies for both players, so have well-defined cost value functions determining the best cost as a function of control-states and clock valuation. They prove that these value functions are in fact piecewise-linear. Hansen et al. [28] later show that the piecewise-linear description has only polynomially many line segments and can be computed in polynomial time[2]. We derive the following lemma.

▶ **Lemma 9.** *A qf-LA formula expressing Vanish_X is effectively computable in polynomial time. More precisely, there is a set $\mathcal{I}$ of polynomially many consecutive intervals $\{a_0\}(a_0,a_1)\{a_1\}(a_1,a_2)\{a_2\},\dots(a_k,\infty)$ so that*

$$\mathit{Vanish}_X(x,t) \;\equiv \bigvee_{0\le i\le k} (x = a_i \wedge t \geq_i c_i) \;\vee\; (a_i < x < a_{i+1} \wedge t \geq_i c_i - b_i x),$$

where the $a_i, c_i \in \mathbb{R}$ can be represented using polynomially many bits, $\geq_i \in \{\geq, >\}$ and $b_i \in \{0,1\}$, for all $0 \le i < k$.

Proof (Sketch). One can construct a one-clock priced timed game in which minimizer's strategies correspond to derivation trees. To do this, let unary rules $X \overset{\varphi;R}{\Longrightarrow} Y$ carry over as transitions between (minimizer) states X,Y; vanishing rules $X \overset{\varphi}{\Longrightarrow} \emptyset$ are replaced by transitions leading to a new target state $\bot$, which has a clock-resetting self-loop. Branching rules $X \overset{\varphi;R}{\Longrightarrow} Y + Z$ can be implemented by rules $X \overset{\varphi}{\Longrightarrow} [Y,Z,\varphi]$, $[Y,Z,\varphi] \overset{\varphi;R}{\Longrightarrow} Y$ and $[Y,Z,\varphi] \overset{\varphi;R}{\Longrightarrow} Z$, where X,Y,Z are minimizer states and $[Y,Z,\varphi]$ is a maximizer state. The cost of staying in a state is 1, transitions carry no costs. Moreover, we need to prevent maximizer from elapsing time from the states she controls. For this reason, we consider an extension of price timed games where maximizer cannot elapse time. In the constructed game, minimizer has a strategy to reach $(\bot,0)$ from (X,x) at cost t iff $(X,x) \xrightarrow{t} \emptyset$. The result now follows from [28, Theorem 4.11] (with minor adaptations in order to consider the more restrictive case where maximizer cannot elapse time) that computes value functions for the cost of reachability in priced timed games. These are piecewise-linear with only polynomially many line segments of slopes 0 or 1 which allows to present Vanish_X in qf-LA as stated.

A more detailed and direct construction can be found in Appendix D. ◀

Lemma 9 allows us to compute a polynomial number of intervals $\mathcal{I}$ sufficient to describe the Vanish_X predicates. We will call a pair $(X,I) \in \mathcal{X} \times \mathcal{I}$ a *region* here. A crucial ingredient for our construction will be timed automata that are restricted in which regions they are allowed to produce as side-effects. To simplify notations let us assume w.l.o.g. that the given TBPP has no resets along branching rules. Let $I(r) \in \mathcal{I}$ denote the unique interval containing

[2] This observation was already made, without proof, in [43, Sec. 7.2.2].

$r \in \mathbb{R}_{\geq 0}$, and for a subset $S \subseteq \mathcal{X} \times \mathcal{I}$ of regions write "$Z \in S$" for the clock constraint expressing that $(Z, I(x)) \in S$. More precisely,

$$Z \in S \;\equiv \bigvee_{(Z,(a_i,a_{i+1}))\in S} a_i < x < a_{i+1} \;\vee \bigvee_{(Z,\{a_i\}))\in S} a_i = x.$$

▶ **Definition 10.** *Let $S \subseteq \mathcal{X} \times \mathcal{I}$ be a set of regions for the 1-TBPP $(\{x\}, \mathcal{X}, \mathcal{R})$. We define a timed automaton $TA_S = (\{x\}, \mathcal{X}, \mathcal{R}_S)$ so that $\mathcal{R}_S$ contains all of the rules in $\mathcal{R}$ with rhs of size 1 and none of the vanishing rules. Moreover, every branching rule $X \xRightarrow{\varphi} Y + Z$ in $\mathcal{R}$ introduces*

- *a rule $X \xRightarrow{\varphi'} Y$ guarded by $\varphi' \stackrel{def}{=} \varphi \wedge Z \in S$, and*
- *a rule $X \xRightarrow{\varphi'} Z$ guarded by $\varphi' \stackrel{def}{=} \varphi \wedge Y \in S$.*

▶ **Theorem 11.** *The reachability problem for 1-clock TBPP is in* NP.

Proof. Suppose that there is a derivation tree witnessing a positive instance of the reachability problem and so that all branches leading to targets have duration τ. We can represent a node by a triple $(A, a, \hat{a}) \in (\mathcal{X} \times \mathbb{R} \times \mathbb{R})$, where (A, a) is a TBPP process and the third component $\hat{a}$ is the total time elapsed so far. Call a node $(A, a, \hat{a})$ *productive* if it lies on a branch from root to some target node. Naturally, every node $(A, a, \hat{a})$ has a unique region $(A, I(a))$ associated with it. For a productive node let us write

$$S(A, a, \hat{a})$$

for the set of regions of nodes which are descendants of $(A, a, \hat{a})$ which are non-productive but have a productive parent. See Figure 2 (left) for an illustration. Observe that

1. The sets $S(A, a, \hat{a})$ can only decrease along a branch from root to a target.
2. If $(A, a, \hat{a})$ has a productive descendant $(C, c, \hat{c})$ such that $S(A, a, \hat{a}) = S(C, c, \hat{c})$, then $(A, a) \xrightarrow{\hat{c}-\hat{a}} (C, c)$ in the timed automaton TA_S.
3. Suppose $(A, a, \hat{a})$ has only one productive child $(C, c, \hat{c})$ and that $S(A, a, \hat{a}) \supset S(C, c, \hat{c})$. Then it must also have another child $(B, b, \hat{b})$ s.t. $Vanish_B(b, \tau - \hat{b})$ holds.

The first two conditions are immediate from the definitions of S and TA_S. To see the third, note that the $S(A, a, \hat{a}) \supset S(C, c, \hat{c})$ implies that $(A, a, \hat{a})$ has some non-productive descendant $(B, b, \hat{b})$ whose region $(B, I(b))$ is not in $S(C, c, \hat{c})$. Since $(C, c, \hat{c})$ is the only productive child, that descendant must already be a child of $(A, a, \hat{a})$. Finally, observe that every non-productive node $(B, b, \hat{b})$ satisfies $Vanish_B(b, \tau - \hat{b})$, as otherwise one of its descendants is present at time τ, and thus must be a target node, contradicting the non-productivity assumption.

The conditions above allow us to use labelled trees of polynomial size as reachability witnesses: These witnesses are labelled trees as above where only as polynomial number of checkpoints along branches from root to target are kept: A checkpoint is either the least common ancestor of two target nodes (in which case a corresponding branching rule must exist), or otherwise it is a triple of nodes as described by condition (3), where a region $(B, I(b))$ is produced for the last time. The remaining paths between checkpoints are positive reachability instances of timed automata TA_S, as in condition (2), where the bottom-most automata TA_S satisfy that if $(U, I) \in S$ then $(z, 0) \in Vanish_U$ for all $z \in I$. Cf. Figure 2 (right). Notice that the existence of a witness of this form is expressible as a polynomially large ∃LA formula thanks to Lemma 9 and Theorem 2.

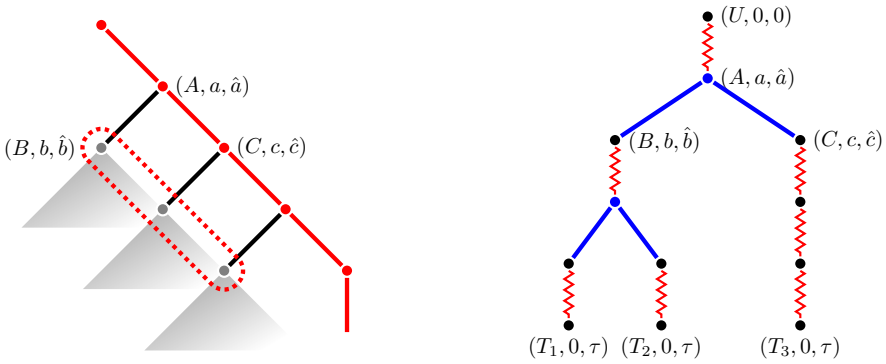

Figure 2 Left: Nodes on the red branch are productive, grey sub-trees are non-productive. $S(A, a, \hat{a})$ contains the regions of nodes in the dotted region. It holds that $S(A, a, \hat{a}) \supseteq S(C, c, \hat{c})$ and the inequality is strict iff $(B, I(b)) \in S(C, c, \hat{c})$. **Right:** small reachability witnesses contain checkpoint where two productive branches split (in blue) or where the allowed side-effects S strictly decrease (red). The intermediate paths are runs of S-restricted TA.

Clearly, every full derivation tree gives rise to a witness of this form. Conversely, assume a witness tree as above exists. One can build a partial derivation tree by unfolding all intermediate TA paths between consecutive checkpoints. It remains to show that whenever some TA_S uses a rule $A \overset{\varphi}{\Longrightarrow} C$ originating from a TBPP rule $A \overset{\varphi}{\Longrightarrow} B + C$ to produce a productive node $(C, c, \hat{c})$ then the node $(B, b, \hat{b})$ produced as side-effect can vanish in time $\tau - \hat{b}$, i.e., we have to show that then $\mathit{Vanish}_B(b, \tau - \hat{b})$.

W.l.o.g. let $\mathit{Vanish}_B(x, t) \equiv t \geq d - xf$ (the case with $>$ is analogous) for some $d \in \mathbb{R}_{\geq 0}$ and $f \in \{0, 1\}$. Observe that the region $(B, I(b))$ is in S by definition of TA_S and that the witness contains a later node $(B, b', \hat{b}')$ with $\mathit{Vanish}_B(b', \tau - \hat{b}')$, and thus

$$\tau - \hat{b}' \geq d - b'f.$$

Notice also that $b' \geq b + \hat{b}' - \hat{b}$ as in the worst-case no reset appears on the path between the parent of $(B, b, \hat{b})$ and $(B, b', \hat{b}')$. Together with the inequality above we derive that $\tau - \hat{b} \geq d - bf$, meaning that indeed $\mathit{Vanish}_B(b, \tau - \hat{b})$ holds, as required. ◀

5 Multi-Clock TBPP

In this section we consider the complexities of coverability and reachability problems for TBPP with multiple clocks. For the upper bounds we will reduce to the reachability problem for TA [7] and to solving reachability games for TA [31].

▶ **Theorem 12.** *The coverability problem for k-TBPP with $k \geq 2$ clocks is* PSPACE*-complete.*

Proof. The lower bound already holds for the reachability problem of 2-clock TA [24] and hence for the *simple* TBPP coverability. For the upper bound, consider an instance where $\mathcal{A} = (\mathcal{C}, \mathcal{X}, \mathcal{R})$ is a k-TBPP and $T_1, \ldots, T_m$ are the target nonterminals. We reduce to the reachability problem for TA $\mathcal{B} = (\mathcal{C}', \mathcal{X}', \mathcal{R}')$ with exponentially many control states $\mathcal{X}'$, but only $|\mathcal{C}'| = O(k \cdot |\mathcal{X}| \cdot m)$ many clocks. The result then follows by the classical region

construction of [7], which requires space logarithmic in the number of nonterminals and polynomial in the number of clocks. The main idea of this construction is to introduce (exponentially many) new nonterminals and rules to simulate the original behaviour on bounded configurations only.

Let $n = m + 2$. We have a clock $x_{X,i} \in \mathcal{C}'$ for every original clock $x \in \mathcal{C}$, nonterminal $X \in \mathcal{X}$, and index $1 \leq i \leq n$, and a nonterminal of the form $[\alpha] \in \mathcal{R}'$ for every multiset $\alpha \in \mathcal{X}^{\oplus}$ of size at most $|\alpha| \leq n$. Since we are solving the coverability problem, we do not need to address vanishing rules $X \xRightarrow{\varphi;R} \emptyset$ in $\mathcal{R}$, which are ignored. We will use clock assignments $S_{X,i} \equiv \bigwedge_{i \leq j \leq n-1} x_{X,j} := x_{X,j+1}$ shifting by one position the clocks corresponding to occurrences $j = i, i+1, \ldots, n-1$ of X. We have three families of rules:

1. **(unary rules).** For each rule $X \xRightarrow{\varphi;R} Y$ in $\mathcal{R}$ and multiset $\beta \in \mathcal{X}^{\oplus}$ of the form $\beta = \gamma + X + \delta$ of size $|\beta| \leq n$, for some $\gamma, \delta \in \mathcal{X}^{\oplus}$, we have a corresponding rule in $\mathcal{R}'$

$$[\beta] \xRightarrow{\varphi|_{X,i};\, R|_{X,i;Y,j};\, S_{X,i}} [\gamma + Y + \delta]$$

   for every occurrence $1 \leq i \leq \beta(X)$ of X in β and for $j = \beta(Y) + 1$, where $\varphi|_{X,i}$ is obtained from φ by replacing each clock x with $x_{X,i}$, and $R|_{X,i;Y,j}$ is obtained from R by replacing every assignment $x := y$ by $x_{Y,j} := x_{X,i}$, and $x := 0$ by $x_{Y,j} := 0$.
2. **(branching rules).** Let $X \Longrightarrow Y + Z$ in $\mathcal{R}$ be a branching rule. We assume w.l.o.g. that it has no tests and no assignments, and that X, Y, Z are pairwise distinct. We add rules in $\mathcal{R}'$

$$[\alpha + X] \xRightarrow{R;S_{X,i}} [\beta], \quad \text{with } \beta = \alpha + Y + Z \text{ and } |\beta| \leq n,$$

   for all $1 \leq i \leq \alpha(X)$ and $\alpha \in \mathcal{X}^{\oplus}$, where $R \equiv \bigwedge_{x \in \mathcal{C}} x_{Y,\beta(Y)} := x_{X,i} \wedge x_{Z,\beta(Z)} := x_{X,i}$ copies each clock $x_{X,i}$ into $x_{Y,\beta(Y)}$ and $x_{Z,\beta(Z)}$, and $S_{X,i}$ was defined earlier.
3. **(shrinking rules).** We also add rules that remove unnecessary nonterminals: For every $\beta = \alpha + X \in \mathcal{X}^{\oplus}$ with $|\beta| \leq n$ and index $1 \leq i \leq \beta(X)$ denoting which occurrence of X in β we want to remove, we have a rule $[\alpha + X] \xRightarrow{S_i} [\alpha]$ in $\mathcal{R}'$.

It remains to argue that $(X, \vec{0}) \xrightarrow{*} (T_1, \vec{0}) + \cdots + (T_n, \vec{0})$ in $\mathcal{A}$ if, and only if, $([X], \vec{0}) \xrightarrow{*} ([T_1 + \cdots + T_n], \vec{0})$ in $\mathcal{B}$. This can be proven via induction on the depth of the derivation tree, where the induction hypothesis is that every configuration α of size at most n can be covered in with a derivation tree of depth d in $\mathcal{A}$ if, and only if, in the timed automaton $\mathcal{B}$ the configuration $[\alpha]$ can be reached via a path of length at most d. ◀

▶ **Theorem 13.** *The reachability problem for TBPP is* EXPTIME*-complete. Moreover,* EXPTIME*-hardness already holds for k-TBPP emptiness, if 1) $k \geq 2$ is any fixed number of clocks, or 2) k is part of the input but only* 0 *or* 1 *appear as constants in clock constraints.*

In the remainder of this section contains a proof of this result, in three steps: In the first step (Lemma 14) we show an EXPTIME upper bound for the special case of *simple reachability*, i.e., when the target configuration has size 1. As a second step (Lemma 15) we reduce general case to simple reachability and thereby prove the upper bound claimed in Theorem 13. As a third step (Lemma 16), we prove the corresponding lower bound.

▶ **Lemma 14** (Simple reachability)**.** *The simple reachability problem for TBPP is in* EXPTIME*. More precisely, the complexity is exponential in the number of clocks and the maximal clock constant, and polynomial in the number of nonterminals.*

Proof (Sketch). We reduce to TA reachability games, where two players (Min and Max) alternatingly determine a path of a TA, by letting the player who owns the current nonterminal pick time elapse and a valid successor configuration. Min and Max aim to minimize/maximize the time until the play first visits a target nonterminal T. TA reachability games can be solved in EXPTIME, with the precise time complexity claimed above [31, Theorem 5]. The idea of the construction is to let Min produce a derivation tree along the branch that leads to (unique) target process. Whenever she proposes to go along branching rule, Max gets to claim that the other sibling, not on the main branch, cannot be removed until the main branch ends. This can be faithfully implemented by storing only the current configuration on the main branch plus one more configuration (of Max's choosing) that takes the longest time to vanish. Min can develop both independently but must apply time delays to both simultaneously. Min wins the game if she can reach the target nonterminal and before that moment all the other branches have vanished. ◀

▶ **Lemma 15.** *The reachability problem for TBPP is in* EXPTIME.

Proof. First notice that the special case of reachability of the empty target set trivially reduces to the simple reachability problem by adding a dummy nonterminal, which is created once at the beginning and has to be the only one left at the end. Suppose we have an instance of the k-TBPP reachability problem with target nonterminals $T_1, T_2, \ldots, T_m$. We will create an instance of simple reachability where the number of nonterminals increases exponentially but the number of clocks is $O(k \cdot |\mathcal{X}| \cdot m)$. In both cases, the claim follows from Lemma 14.

We introduce a nonterminal $[\beta]$ for every multiset $\beta \in \mathcal{X}^{\oplus}$ of size $|\beta| \leq n := m + 2$, and we have the same three family of rules as in proof of Theorem 12, where the last family 3. is replaced by the family below:

3'. We add extra branching rules in order to maintain nonterminals $[\beta]$ corresponding to small multisets $|\beta| \leq n$. Let $\beta \in \mathcal{X}^{\oplus}$ of size $|\beta| \leq n$ and consider a partitioning $\beta = \beta_1 + \beta_2$, for some $\beta_1, \beta_2 \in \mathcal{X}^{\oplus}$. We identify β with the set $\beta = \{(X, i) \mid X \in \mathcal{X}, 1 \leq i \leq \beta(X)\}$ of pairs (X, i), where i denotes the i-th occurrence of X in β (if any), and similarly for β_1, β_2. We add a branching rule

$$[\beta] \Longrightarrow (\beta, f, \beta_1) + (\beta, f, \beta_2),$$

where (β, f, β_i) are intermediate locations, for every bijection $f : \beta \to \beta_1 \cup \beta_2$ assigning an occurrence of X in β to an occurrence of X either in β_1 or β_2. We then have clock reassigning (non-branching) rules

$$(\beta, f, \beta_1) \overset{S_1}{\Longrightarrow} [\beta_1] \quad \text{and} \quad (\beta, f, \beta_2) \overset{S_2}{\Longrightarrow} [\beta_2],$$

where $S_1 \equiv \bigwedge_{x \in \mathcal{C}} \bigwedge_{X \in \mathcal{X}} \bigwedge_{1 \leq i \leq \beta_1(X)} x_{X,i} := x_{f^{-1}(X,i)}$ and similarly for S_2. ◀

▶ **Lemma 16.** *The non-emptiness problem for TBPP is* EXPTIME*-hard already in discrete time, for 1) TBPP with constants in* $\{0, 1\}$ *(where the number of clocks is part of the input), and 2) for* k*-TBPP for every fixed number of clocks* $k \geq 2$*.*

6 Conclusion

We introduced basic parallel processes extended with global time and studied the complexities of several natural decision problems, including variants of the coverability and reachability problems. Table 1 summarizes our findings.

The exact complexity status of the simple reachability problem for 1-TBPP is left open. An NP upper bound holds from the (general) reachability problem (by Theorem 11) and PTIME-hardness comes from the emptiness problem for context-free grammars. We conjecture that a matching polynomial-time upper bound holds.

Also left open for future work are *succinct* versions the coverability and reachability problems, where the target size is given in binary. A reduction from subset-sum games [24] shows that the succinct coverability problem for 1-TBPP is PSPACE-hard. This implies that our technique showing the NP-membership for the non-succinct version of the coverability problem (cf. Theorem 7) does not extend to the succinct variant, and new ideas are needed.

	Emptiness	Simple Coverability	Coverability	Simple Reachability	Reachability
TBPP	EXPTIME [Lem 16], [31]	PSPACE [Thm 12]	PSPACE [Thm 12]	EXPTIME [Thm 14]	EXPTIME [Thm 13]
1-TBPP	PTIME [34]	NL [34]	NP [Thm 7]	PTIME / NP	NP [Thm 11]

Table 1 Results on TBPP and 1-clock TBPP. The decision problems are complete for the stated complexity class. Simple Coverability/Reachability refer to the variants where the target has size 1.

References

1 P. A. Abdulla, M. F. Atig, and J. Cederberg. Timed lossy channel systems. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, 2012.

2 P. A. Abdulla, M. F. Atig, R. Ciobanu, R. Mayr, and P. Totzke. Universal Safety for Timed Petri Nets is PSPACE-complete. In *International Conference on Concurrency Theory (CONCUR)*, 2018.

3 P. A. Abdulla, M. F. Atig, and S. N. Krishna. Perfect timed communication is hard. In *International Conference on Formal Modeling and Analysis of Timed Systems (FORMATS)*, 2018.

4 P. A. Abdulla, M. F. Atig, and J. Stenman. Dense-timed pushdown automata. In *ACM/IEEE Symposium on Logic in Computer Science (LICS)*, 2012.

5 P. A. Abdulla and B. Jonsson. Model checking of systems with many identical timed processes. *Theoretical Computer Science*, 2003.

6 P. A. Abdulla and A. Nylén. Timed Petri Nets and BQOs. In *Applications and Theory of Petri Nets and Concurrency (PETRI NETS)*, 2001.

7 R. Alur and D. L. Dill. A theory of timed automata. *Theoretical Computer Science*, 1994.

8 M. Benerecetti, S. Minopoli, and A. Peron. Analysis of timed recursive state machines. In *International Symposium on Temporal Representation and Reasoning (TIME)*, 2010.

9 M. Benerecetti and A. Peron. Timed recursive state machines: Expressiveness and complexity. *Theoretical Computer Science*, 2016.

10 B. Bérard, A. Labroue, and P. Schnoebelen. Verifying performance equivalence for timed basic parallel processes. In *International Conference on Foundations of Software Science and Computational Structures (FoSSaCS)*, 2000.

11 B. Boigelot, S. Jodogne, and P. Wolper. An effective decision procedure for linear arithmetic over the integers and reals. *ACM Trans. Comput. Logic*, 2005.

12 A. Bouajjani, R. Echahed, and R. Robbana. On the automatic verification of systems with continuous variables and unbounded discrete data structures. In *Hybrid Systems (HS)*, 1994.

13 P. Bouyer, K. G. Larsen, N. Markey, and J. I. Rasmussen. Almost optimal strategies in one clock priced timed games. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, 2006.

14 S. Christensen. *Decidability and Decomposition in Process Algebras*. PhD thesis, School of Informatics, University of Edinburgh, 1993.

15 L. Clemente. Decidability of Timed Communicating Automata. *arXiv e-prints*, 2018.

16 L. Clemente, F. Herbreteau, A. Stainer, and G. Sutre. Reachability of communicating timed processes. In *International Conference on Foundations of Software Science and Computational Structures (FoSSaCS)*, 2013.

17 L. Clemente and S. Lasota. Binary reachability of timed pushdown automata via quantifier elimination. In *International Colloquium on Automata, Languages and Programming (ICALP)*, 2018.

18 L. Clemente, S. Lasota, R. Lazić, and F. Mazowiecki. Timed pushdown automata and branching vector addition systems. In *ACM/IEEE Symposium on Logic in Computer Science (LICS)*, 2017.

19 H. Comon and Y. Jurski. Timed automata and the theory of real numbers. In *International Conference on Concurrency Theory (CONCUR)*, 1999.

20 Z. Dang. Binary reachability analysis of pushdown timed automata with dense clocks. In *Computer Aided Verification (CAV)*, 2001.

21 C. Dima. Computing reachability relations in timed automata. In *ACM/IEEE Symposium on Logic in Computer Science (LICS)*, 2002.

22 C. Dima. A class of automata for computing reachability relations in timed systems. In *Verification of Infinite State Systems with Applications to Security (VISSAS)*, 2005.

23 J. Esparza. Petri Nets, Commutative Context-Free Grammars, and Basic Parallel Processes. *Fundamenta Informaticae*, 1997.

24 J. Fearnley and M. Jurdziński. Reachability in two-clock timed automata is pspace-complete. *Information and Computation*, 2015.

25 J. Ferrante and C. Rackoff. A decision procedure for the first order theory of real addition with order. *SIAM Journal on Computing*, 1975.

26 M. Fränzle, K. Quaas, M. Shirmohammadi, and J. Worrell. Effective definability of the reachability relation in timed automata. *arXiv e-prints*, 2019.

27 S. Haddad, S. Schmitz, and Ph. Schnoebelen. The ordinal recursive complexity of timed-arc Petri nets, data nets, and other enriched nets. In *ACM/IEEE Symposium on Logic in Computer Science (LICS)*, 2012.

28 T. D. Hansen, R. Ibsen-Jensen, and P. B. Miltersen. A faster algorithm for solving one-clock priced timed games. In *International Conference on Concurrency Theory (CONCUR)*, 2013.

29 A. Heußner, J. Leroux, A. Muscholl, and G. Sutre. Reachability analysis of communicating pushdown systems. *LMCS*, 2012.

30 M. Jurdzinski, F. Laroussinie, and J. Sproston. Model Checking Probabilistic Timed Automata with One or Two Clocks. *Logical Methods in Computer Science*, 2008.

31 M. Jurdziński and A. Trivedi. Reachability-time games on timed automata. In *International Colloquium on Automata, Languages and Programming (ICALP)*, 2007.

32 P. Krčál and R. Pelánek. On sampled semantics of timed systems. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, 2005.

33 P. Krcal and W. Yi. Communicating timed automata: the more synchronous, the more difficult to verify. In *Computer Aided Verification (CAV)*, 2006.

34 F. Laroussinie, N. Markey, and P. Schnoebelen. Model checking timed automata with one or two clocks. In *International Conference on Concurrency Theory (CONCUR)*, 2004.

35 K. G. Larsen, P. Pettersson, and W. Yi. Uppaal in a nutshell. *International Journal on Software Tools for Technology Transfer*, 1997.

36 S. Lasota and I. Walukiewicz. Alternating timed automata. *ACM Trans. Comput. Logic*, 2008.

37 P. M. Merlin. *A Study of the Recoverability of Computing Systems*. PhD thesis, University of California, Irvine, 1974.

38 J. Ouaknine and J. Worrell. On the decidability of metric temporal logic. In *ACM/IEEE Symposium on Logic in Computer Science (LICS)*, 2005.

39 L. Popova. On Time Petri Nets. *Journal of Information Processing and Cybernetics*, 1991.

40 M. Presburger. Über die Vollständigkeit eines gewissen Systems der Arithmetik ganzer Zahlen, in welchem die Addition als einzige Operation hervortritt. *Comptes Rendus Premier Congrès des Mathématicienes des Pays Slaves*, 1930.

41 K. Quaas. Verification for Timed Automata extended with Unbounded Discrete Data Structures. *Logical Methods in Computer Science*, 2015.

42 V. V. Ruiz, F. C. Gomez, and D. d. F. Escrig. On Non-Decidability of Reachability for Timed-Arc Petri Nets. In *International Workshop on Petri Nets and Performance Models (PNPM)*, 1999.

43 A. Trivedi. *Competative Optimisation on Timed Automata*. PhD thesis, University of Warwick, 2009.

44 A. Trivedi and D. Wojtczak. Recursive Timed Automata. In *International Symposium on Automated Technology for Verification and Analysis (ATVA)*, 2010.

45 K. N. Verma, H. Seidl, and T. Schwentick. On the complexity of equational Horn clauses. In *International Conference on Automated Deduction (CADE)*, 2005.

46 V. Weispfenning. Mixed real-integer linear quantifier elimination. In *International Symposium on Symbolic and Algebraic Computation (ISSAC)*, 1999.

47 S. Yovine. Kronos: a verification tool for real-time systems. *International Journal on Software Tools for Technology Transfer*, 1997.

A Missing proofs for Section 3

▶ **Theorem 1.** *The existential fragment* $\exists LA$ *of* LA *is in* NP.

It was proved in [46] that LA admits quantifier elimination by reduction to quantifier elimination procedures for PA and RA. However, the complexity of deciding the existential fragment of LA was not discussed, and moreover the procedure translating a formula of LA to the separated fragment has an exponential blow-up when dealing with modulo constraints in the proof of [46, Theorem 3.1] (case 2), and moreover it relies on an intermediate transformation with no complexity bound [46, Lemma 3.3]. Our result was obtained independently from [11], where a similar argument is given showing that each sentence of LA can be transformed in polynomial time into a logically equivalent boolean combination of PA and RA sentences. In fact, investigating the proof of [11, Theorem 3.1] one can see that their translation even preserves existential formulas. We give below a short self-contained argument reducing LA to PA and RA with only a polynomial blowup, with the further property that it preserves existential formulas.

Proof. By introducing linearly many new existentially quantified variables and suitable defining equalities, we assume w.l.o.g. that terms are *shallow*:

$$s, t \ ::= \ x \mid k \mid \lfloor x \rfloor \mid \mathit{frac}(x) \mid -x \mid x + y \mid k \cdot x.$$

Since now we have atomic propositions of the form $s \leq t$ with s, t shallow terms, we assume that we have no terms of the form "$-x$" (by moving it to the other side of the relation and possibility introducing a new existential variable to make the term shallow again). Moreover, we can also eliminate terms of the form $k \cdot x$ by introducing $O(\log k)$ new existential variables and using iterated doubling (based on the binary expansion of k); for instance, $5 \cdot x$ is

replaced by $x_0 + x_2$, by adding new variables $x_0, \ldots, x_2$ and equalities $x_0 = x$, $x_1 = x_0 + x_0$, $x_2 = x_1 + x_1$. We end up with the following further restricted syntax of terms:

$$s, t \ ::= \ x \mid k \mid \lfloor x \rfloor \mid \mathit{frac}(x) \mid x + y.$$

We can now expand atomic propositions by replacing the expression on the left with the equivalent one on the right:

$$s \leq t \qquad \text{iff} \qquad \lfloor s \rfloor < \lfloor t \rfloor \vee (\lfloor s \rfloor = \lfloor t \rfloor \wedge \mathit{frac}(s) \leq \mathit{frac}(t)),$$

We push the integral $\lfloor \_ \rfloor$ and fractional $\mathit{frac}(\_)$ operations inside terms, according to the following rules:

$$\begin{array}{rr} \lfloor k \rfloor \to k & \mathit{frac}(k) \to 0 \\ \lfloor \lfloor x \rfloor \rfloor \to \lfloor x \rfloor & \mathit{frac}(\lfloor x \rfloor) \to 0 \\ \lfloor \mathit{frac}(x) \rfloor \to 0 & \mathit{frac}(\mathit{frac}(x)) \to \mathit{frac}(x) \end{array}$$

It remains to consider sums $x + y$. We perform a case analysis on $\mathit{frac}(x) + \mathit{frac}(y)$:

$$\begin{array}{rl} s \sim \lfloor x + y \rfloor \to & (s \sim \lfloor x \rfloor + \lfloor y \rfloor \wedge \mathit{frac}(x) + \mathit{frac}(y) < 1) \vee \\ & (s \sim \lfloor x \rfloor + \lfloor y \rfloor + 1 \wedge \mathit{frac}(x) + \mathit{frac}(y) \geq 1), \\ s \sim \mathit{frac}(x + y) \to & (s \sim \mathit{frac}(x) + \mathit{frac}(y) \wedge \mathit{frac}(x) + \mathit{frac}(y) < 1) \vee \\ & (s \sim \mathit{frac}(x) + \mathit{frac}(y) - 1 \wedge \mathit{frac}(x) + \mathit{frac}(y) \geq 1). \end{array}$$

We thus obtain a logically equivalent *separated formula*, i.e., one where integral $\lfloor x \rfloor$ and fractional $\mathit{frac}(y)$ variables never appear together in the same term. Since we only added existentially quantified variables in the process, the resulting formula is still in the existential fragment, which can be decided in NP by calling separately decision procedures for PA and RA. ◀

▶ **Lemma 3.** *For every configurations (X, u) and (Y, v) of $\mathcal{A}$ and total time elapse $\delta \geq 0$,*

$$u \xrightarrow{\delta}_{XY} v \quad \textit{iff} \quad (u, \delta, v) \models \varphi_{XY}(x, t, x').$$

Proof. Consider the following binary relation R between the configurations of the infinite transition system induced by the TA $\mathcal{A}$ and states of the NFA $\mathcal{B}$:

$$((X, u), (X', \lambda)) \in R \quad \text{iff} \quad X = X' \wedge u \in \lambda. \tag{1}$$

Let $\hat{b} = r$ for $b = (r, a)$ and $\hat{\tau} = \tau$ otherwise. It is immediate to show that R is a variant of timed-abstract bisimulation, in the following sense: For every configuration c of $[\![\mathcal{A}]\!]$ and state d of $\mathcal{B}$, if $(c, d) \in R$, then

1. For every transition of $\mathcal{A}$ of the form $c \xrightarrow{r} c'$ there is a transition of $\mathcal{B}$ of the form $d \xrightarrow{b} d'$ s.t. $\hat{b} = r$ and $(c', d') \in R$. Moreover, if $r = \tau$ is a time elapse, then in fact $d \xrightarrow{\tau} d'$ *for every* d' s.t. $(c', d') \in R$.
2. For every transition of $\mathcal{B}$ of the form $d \xrightarrow{b} d'$ there is a transition of $[\![\mathcal{A}]\!]$ of the form $c \xrightarrow{r} c'$ s.t. $\hat{b} = r$ and $(c', d') \in R$. Moreover, if $b = \tau$ is a symbolic time elapse, then in fact $c \xrightarrow{\tau} c'$ *for every* c' s.t. $(c', d') \in R$.

The two additional conditions in each point distinguish R from timed-abstract bisimulation.

For the "only if" direction, assume $u \xrightarrow{\delta}_{XY} v$, as witnessed by a path

$$(X_0, u_0) \xrightarrow{r_1\delta_1} (X_1, u_1) \xrightarrow{r_2\delta_2} \cdots \xrightarrow{r_n\delta_n} (X_n, u_n),$$

with $(X_0, u_0) = (X, u)$ and $(X_n, u_n) = (Y, v)$, where u, v are the initial, resp., final values of the clock, and $\delta = \delta_0 + \cdots + \delta_n$ is the total time elapsed. Let $\lambda, \mu \in \Lambda$ be the unique intervals s.t. $u \in \lambda, v \in \mu$, and take $c = (X, \lambda), d = (Y, \mu)$. Since $((X, u), c) \in R$, by the definition of R there exists a corresponding path in $\mathcal{B}$

$$c_0 \xrightarrow{b_1\tau} c_1 \xrightarrow{b_2\tau} \cdots \xrightarrow{b_n\tau} c_n,$$

where $c_0 = (X, \lambda)$, $c_n = (Y, \mu)$, for every $1 \leq i \leq n$, $((X_i, u_i), c_i) \in R$, c_i of the form (X_i, λ_i), and b_i is of the form (r_i, a_i), with $a_i = \checkmark_{\lambda_i}$ whenever r_i is a reset (and $a_i = \varepsilon$ otherwise). Let $r_{i_1}, \ldots, r_{i_m}$ be all reset transitions in $r_1, \ldots, r_n$. The time elapsed between the j-th and the $(j+1)$-th reset is $u_{i_{j+1}-1} = \delta_{i_j} + \delta_{i_j+1} + \cdots + \delta_{i_{j+1}-1}$, and, by the invariant above, $u_{i_{j+1}-1} \in \lambda_{i_{j+1}-1}$ and $a_{i_{j+1}-1} = \checkmark_{\lambda_{i_{j+1}-1}}$. Consequently, the total time elapses is $\delta = (u_{i_1-1} - u_0) + (u_{i_2-1} + \cdots + u_{i_m-1}) + u_m$. By the definition of $\varphi_{L_{cd}}$, this shows $(u, \delta, v) \models x \in \lambda \wedge x' \in \mu \wedge \psi_{cd}(x, t, x')$, as required.

For the "if" direction, assume $(u, \delta, v) \models \varphi_{XY}(x, t, x')$. There exist configurations $c = (X, \lambda)$ and $d = (Y, \mu)$ s.t. $u \in \lambda$, $v \in \mu$, and $(u, \delta, v) \models \psi_{cd}(x, t, x')$. By identifying variables with their values to simplify the notation in the sequel, there exist $\vec{y} = (y_\lambda)_{\lambda\in\Lambda}$ s.t. $\vec{y} \models \psi_{L_{cd}}$ and $\vec{x} = (x_\lambda)_{\lambda\in\Lambda}$ s.t. $x_\lambda \in y_\lambda \cdot \lambda$ and $\delta = v - u + \sum_{\lambda\in\Lambda} x_\lambda$. By the definition of $\psi_{L_{cd}}$, there exists a run in $\mathcal{B}$ from c to d of the form

$$(X, \lambda) \xrightarrow{\tau(r_1,a_1)} (X_1, \lambda_1) \xrightarrow{\tau(r_2,a_2)} \cdots \xrightarrow{\tau(r_n,a_n)} (Y, \mu),$$

where without loss of generality we have composed possibly several τ-transitions together into a single τ-transition, $\overline{\tau}$, whenever r_i is a reset, $a_i = \checkmark_{\lambda_i}$ (and $a_i = \varepsilon$ otherwise). Consequently, y_λ is precisely the number of r_i's s.t. $a_i = \checkmark_\lambda$. Since $((X, u), c) \in R$ and R is a timed-abstract bisimulation, there exists a corresponding run in $[\![\mathcal{A}]\!]$ from (X, u) to (Y, v) of the form

$$(X, u) \xrightarrow{\delta_1 r_1} (X_1, u_1) \xrightarrow{\delta_2 r_2} \cdots \xrightarrow{\delta_n r_n} (Y, v)$$

s.t. $((X_i, u_i), c_i) \in R$ for every $1 \leq i < n$, i.e., $u_i \in \lambda_i$. It remains to show that δ is the total time elapsed by the run above, i.e., $\delta = \delta_1 + \cdots + \delta_n$. Let $r_{i_1}, \ldots, r_{i_m}$ be all reset transitions from $r_1, \ldots, r_n$. By the definition of R, we can choose the u_i's arbitrarily in λ_i. In particular we can choose $u_{i_1} \in \lambda_{i_1}, \ldots, u_{i_m} \in \lambda_{i_m}$ at the time of resets s.t. for every interval $\lambda \in \Lambda$, the cumulative value of the clock at the times of reset when it was in interval λ is precisely $x_\lambda = \sum\{u_{i_j} \mid \lambda_{i_j} = \lambda\}$. Consequently, the total time accumulated by the clock in any reset is $\sum_{\lambda\in\Lambda} x_\lambda = u_{i_1} + \cdots + u_{i_m}$. Moreover, we have $u_{i_1} = \delta_1 + \cdots + \delta_{i_1} - u$, $u_{i_j} = \delta_{i_{j-1}+1} + \cdots + \delta_{i_k}$ for $1 < j \leq m$, and $v = \delta_{i_m+1} + \cdots + \delta_n$. Consequently, $\delta_1 + \cdots + \delta_n = u - v + \sum_{\lambda\in\Lambda} x_\lambda$, as required. ◀

B Missing proofs for Section 5

▶ **Lemma 14** (Simple reachability). *The simple reachability problem for TBPP is in* EXPTIME. *More precisely, the complexity is exponential in the number of clocks and the maximal clock constant, and polynomial in the number of nonterminals.*

Proof. Let $\mathcal{B} = (\mathcal{C}, \mathcal{X}, \mathcal{R})$ be the input TBPP and suppose we want to solve a simple reachability query $(X, \vec{0}) \xrightarrow{*} (T, \vec{0})$ We denote with $\mathcal{X}_\emptyset$ the set $\mathcal{X} \cup \{\emptyset\}$. We construct the TA reachability game $\mathcal{A} = (\mathcal{D}, \mathcal{Y} = \mathcal{Y}_{\min} \cup \mathcal{Y}_{\max}, \mathcal{S})$, where the set of clocks $\mathcal{D}$ contains two copies x_L and x_R for every clock $x \in \mathcal{C}$, locations in $\mathcal{Y}_{\min} = \mathcal{X} \times \mathcal{X}_{\{\emptyset\}}$ are of the form $(X, Y_\emptyset)$ with $Y_\emptyset = Y \in \mathcal{X}$ or $Y_\emptyset = \emptyset$, and $\mathcal{Y}_{\max} = \mathcal{X} \times \mathcal{X} \times \mathcal{X}_\emptyset$. For a formula φ and $D \in \{L, R\}$, we write $\varphi|_D$ for the formula obtained from φ by replacing every clock $x \in \mathcal{C}$ by its copy x_D; similarly for $S|_D$, where S is a sequence of clock assignments. The initial location is $(X, \emptyset)$ and the target location is $(T, \emptyset)$.

The idea of the construction is to let Min stepwise produce a derivation tree along the branch that leads to (unique) target process. Whenever she proposes to go along a branching rule, Max gets to claim that the other sibling, not on the main branch, cannot be removed until the branch ends. This can be faithfully implemented by storing only the current configuration on the main branch plus one more configuration (of Max's choosing) that takes the longest time to vanish. Min can develop both independently but must apply time delays to both simultaneously. The game ends and Min wins when $(T, \emptyset)$ is reached.

A round of the game from position $(X, Y_\emptyset)$ is played as follows. We assume w.l.o.g. that there are only branching and vanishing rules. There are two cases.

1. In the first case, Min plays from the first component. For every rule $X \xRightarrow{\varphi;S} Z_0 + Z_1$, Min chooses that Z_i is the nonterminal which will reach T, and that Z_{1-i} should vanish. Thus, the game has two transitions

$$X \xRightarrow{\varphi|_L; S|_L} (Z_i, Z_{1-i}, Y_\emptyset), \quad \text{for } i \in \{0, 1\},$$

   This is followed by Max, who chooses whether to keep the second component $Y_\emptyset$, or to replace it by Z_{1-i}. Thus, there are transitions

$$(Z_i, Z_{1-i}, Y_\emptyset) \Longrightarrow (Z_i, Y_\emptyset) \quad \text{and} \quad (Z_i, Z_{1-i}, Y_\emptyset) \Longrightarrow (Z_i, Z_{1-i}), \quad \text{for } i \in \{0, 1\}.$$

2. In the second case, Min plays from the second component $Y_\emptyset = Y \neq \emptyset$. There are two subcases.
   a. Min selects a vanishing rule $Y \xRightarrow{\varphi} \emptyset$. The corresponding transition in the game is

$$(X, Y) \xRightarrow{\varphi|_L} (X, \emptyset).$$

   b. Min selects a branching rule $Y \xRightarrow{\varphi;S} Z_0 + Z_1$. which corresponds to the following transition in the game:

$$(X, Y) \xRightarrow{\varphi|_R; S|_R} (X, Z_0, Z_1).$$

   Then, Max chooses to keep $Z \in \{Z_0, Z_1\}$, corresponding to the following two transitions

$$(X, Z_0, Z_1) \Longrightarrow (X, Z), \quad \text{for } Z \in \{Z_0, Z_1\}.$$

We have that Min has a strategy to reach $(T, \emptyset)$ from $(X, \emptyset)$ in the TA game if, and only if, $(X, \vec{0}) \xrightarrow{*} (T, \vec{0})$ in the TBPP. ◀

▶ **Lemma 16.** *The non-emptiness problem for TBPP is* EXPTIME*-hard already in discrete time, for 1) TBPP with constants in* $\{0, 1\}$ *(where the number of clocks is part of the input), and 2) for* k*-TBPP for every fixed number of clocks* $k \geq 2$*.*

First case. The first case can be shown by reduction from the non-emptiness problem of the language intersection of one context-free grammar and several nondeterministic finite automata since we can reduce to the case above by adding polynomially many clocks. We reduce from the non-emptiness problem of the intersection of one CFG G and several nondeterministic finite automata (NFA) $\mathcal{A}_1, \ldots, \mathcal{A}_n$, which is EXPTIME-hard (this is a folklore result; cf. also [29]). We proceed as in a similar reduction for densely-timed pushdown automata [4]. We assume w.l.o.g. that all $\mathcal{A}_i$'s control locations are from a fixed set of integers $Q = \{0, \ldots, n-1\}$. We construct a TBPP $\mathcal{B}$ with $2n+1$ clocks: For each NFA $\mathcal{A}_i$, we have a clock x_i representing its current control location and a clock y_i representing its future control location after the relevant nonterminal reduces to ε; an extra clock t guarantees that no time elapses. We assume w.l.o.g. that G is in Chomsky-Greibach normal form, i.e., productions are either of the form $X \xrightarrow{a} YZ$ or $X \to \varepsilon$.

We now describe the reduction. The simulation of reading input symbol a proceeds in $n+1$ steps. In step 0, for every production $X \xrightarrow{a} YZ$ of G we have a production of $\mathcal{B}$

$$X \xRightarrow{t=0} Y_{a,1}(\vec{x} := \vec{x}, \vec{y} := \vec{k}, t := t) + Z(\vec{x} := \vec{k}, \vec{y} := \vec{y}, t := t).$$

We use here an extended syntax to simplify the presentation. Intuitively, the rule above requires guessing a new tuple of clock values $\vec{k} \in Q^n$, which represent the intermediate control locations of the NFA's after nonterminal Y will vanish. Process $Y_{a,1}$ inherits the value of clocks $\vec{x}$ from X, and the new value of clocks $\vec{y}$ is the guessed $\vec{k}$. Symmetrically, process Z inherits the value of clocks $\vec{y}$ from X, and the new value of clocks $\vec{x}$ is $\vec{k}$. While guessing $\vec{k}$ seemingly requires an exponential number of rules, one can in fact implement it by guessing its components one after the other, by introducing n extra clocks; we avoid the extra bookkeeping for simplicity.

In step i, for $1 \le i \le n$, for every transition $p_i \xrightarrow{a} q_i$ of $\mathcal{A}_i$, we have a production

$$X_{a,i} \xRightarrow{t=0, x_i=p_i, x_i:=q_i} X_{a,i+1}.$$

The current phase ends with a production $X_{a,n+1} \xRightarrow{t=0} X$. Finally, a production $X \to \varepsilon$ of G is simulated by

$$X \xRightarrow{\vec{x}=\vec{y}} \emptyset.$$

The following lemma states the correctness of the construction. We denote by $L_G(X)$ the set of words over the terminal alphabet recognised by nonterminal X in the grammar G, and by $L_{\mathcal{A}_i}(p, q)$, with $p, q \in Q_i$, the set of words s.t. the NFA $\mathcal{A}_i$ has a run from state p to state q.

▶ **Lemma 17.** *We have that $L_G(X) \cap L_{\mathcal{A}_1}(p_1, q_1) \cap \cdots \cap L_{\mathcal{A}_n}(p_n, q_n) \neq \emptyset$ if, and only if, $(X, \mu) \xrightarrow{*} \emptyset$, where $\mu = \{x_1 \mapsto p_1, y_1 \mapsto q_1, \ldots, x_n \mapsto p_n, y_n \mapsto q_n\}$.*

Second case. We now show that the reachability of the empty configuration problem for k-TBPP is EXPTIME-hard for any fixed number of clocks $k \ge 2$. This can be shown by reduction from *countdown games* [30], which are two-player games (Q, T, k) given by a finite set Q of control states, a finite set $T \subseteq (Q \times \mathbb{N}_{>0} \times \mathbb{Q})$ of transitions, labelled by positive integers, and a target number $k \in \mathbb{N}$. All numbers are given in binary encoding. The game is played in rounds, each of which starts in a pair (p, n) where $p \in Q$ and $n \le k$, as follows. First player 0 picks a number $l \le k - n$, so that at least one $(p, l, p') \in T$ exists; Then player 1 picks one such transition and the next round starts in $(p', n + l)$. Player 0 wins iff she can reach a configurations (q, k) for some state q.

Determining the winner in a countdown game is EXPTIME-complete [30] and can easily encoded as a reachability of the empty configuration problem for a 2-TBPP $(\{x_1, x_2\}, Q, \mathcal{R})$. All we need is one nonterminal per control state and two clocks. Suppose $p_1, p_2, \ldots, p_m$ are the l-successors of state p in the game. Then our TBPP has a rule

$$p \xrightarrow{x_1=l; x_1:=0} p_1 + p_2 + \ldots + p_m$$

This checks that the first clock is l and then resets it. Finally, for all states p there is a rule $p \xrightarrow{x_2=k} \emptyset$ that checks if the second clock (which is never reset) equals the target k.

Notice that a derivation tree that reduces $(p, 0, 0)$ to the empty multiset is a winning strategy for player 0 in the countdown game: Since all branches end in a vanishing rule, the total time on the branch was exactly k.

C PSPACE-hardness of 1-TBPP Coverability with Binary Targets

We show that reachability and coverability for 1-clock TBPP with target sets given as a vector in $\mathbb{N}^{\mathcal{X}}$ in binary encoding, is PSPACE-hard. This contrasts with the case of targets encoded in unary, which is NP-complete (Theorem 12).

We reduce from solving subset-sum games, which is PSPACE-complete [24]. A *subset-sum game* (SSG) has as input a natural number $s \in \mathbb{N}$ and a tuple

$$\forall \{u_1, v_1\} \exists \{w_1, z_1\} \cdots \forall \{u_n, v_n\} \exists \{w_n, z_n\},$$

where all $u_i, v_i, w_i, z_i \in \mathbb{N}$ are given in binary. The game is played by two players, $\forall$ and $\exists$ who alternate in choosing numbers: At round i, $\forall$ chooses a number $x_i \in \{u_i, v_i\}$ and $\exists$ chooses a number $y_i \in \{w_i, z_i\}$. At the end of the game, the two players have jointly produced a sequence of numbers $x_1, y_1, \ldots, x_n, y_n$, and $\exists$ wins the game if $x_1 + y_1 + \cdots + x_n + y_n = s$.

Given a SSG as above, we construct a 1-clock TBPP and a target configuration α s.t. $\exists$ wins the SSG iff w is reachable from the initial configuration. $\exists$'s moves are mimicked by branching transitions, and $\forall$'s by nondeterministic choice. An additional process checks that the total time elapsed in every branch is equal to the target sum s.

We can implement the above intuition in a TBPP with a single clock x, nonterminals $\mathcal{X} = \{S, T, F\} \cup \{\forall_i, \exists_i, U_i, V_i \mid 1 \leq i \leq n\} \cup \{\exists_{n+1}\}$ and rules as follows. We have an initial production

$$S \xrightarrow{x=0} \forall_1 + T$$

and a final rule that allows T to check that the total elapsed time is s:

$$T \xrightarrow{x=s; x:=0} F$$

For each round $1 \leq i \leq n$, the i-th move of $\exists$ is modelled by rules

$$\exists_i \xrightarrow{x=w_i; x:=0} \forall_i \quad \text{and} \quad \exists_i \xrightarrow{x=z_i; x=0} \forall_i$$

and the i-th move of $\forall$ by

$$\forall_i \Longrightarrow U_i + V_i, \quad U_i \xrightarrow{x=u_i; x:=0} \exists_{i+1}, \quad \text{and} \quad V_i \xrightarrow{x=v_i; x=0} \exists_{i+1}.$$

An extra rule concludes the simulation: $\exists_{i+1} \xrightarrow{x=0} F$. The target multiset is $\alpha = (2^n + 1) \cdot F$, i.e., $2^n + 1$ copies of F. The correctness of the construction is stated in the lemma below.

▶ **Lemma 18.** *∃ wins the SSG if, and only if,* $(S, \vec{0}) \xrightarrow{*} (\alpha, \vec{0})$.

The reduction above produces a 1-TBPP where $(S, 0) \xrightarrow{*} (\alpha, \vec{0}) + \gamma$ implies that $\gamma = \emptyset$. We thus get the following lower bound.

▶ **Theorem 19.** *The reachability and coverability problems for 1-clock TBPP with target configuration encoded in binary are* PSPACE*-hard.*

D A proof of Lemma 9

▶ **Definition 20.** *An ∃LA formula over variables x, y is* basic *if it is* **true**, **false** *or of form*

$$\phi(x, y) \equiv (a \leq_1 x \leq_2 b) \wedge (c + dx \leq_3 y)$$

for some constants $a, b, c \in \mathbb{R}_{\geq 0}$, $d \in \{-1, 0\}$ and $\leq_1, \leq_2, \leq_3 \in \{<, \leq\}$. That is, its support is the upward-closure (wrt. y) of a finite line segment with end points a, b, and slope 0 or -1. It is crossing *another basic formula $\phi'(x, y) \equiv (a' \leq'_1 x' \leq'_2 b') \wedge (c' + d'x' \leq'_3 y')$ at point $e \in \mathbb{R}$ if $(c + de = c' + d'e) \wedge (a \leq_1 e \leq_2 b) \wedge (a' \leq'_1 e \leq'_2 b') \wedge (d \neq d')$, i.e., their two line segments intersect. A family $\mathcal{F}$ of basic formulae is called* simple *if no two elements cross.*

▶ **Proposition 21.** *If $\phi(x, y)$ is a basic formula then both $\varphi(x, y) \stackrel{def}{=} \exists t.\phi(x + t, y - t)$ and $\varphi(x, y) \stackrel{def}{=} \exists t.\phi(x + t, y)$ are basic formulae.*

▶ **Definition 22.** *Let ϕ be a formula and $\mathcal{F}$ be a family of formulae over variables x, y. We call ϕ* piecewise *in $\mathcal{F}$ if it is equivalent to a finite disjunction of formulae in $\mathcal{F}$. It is called* piecewise basic *if it is the finite disjunction of basic formulae.*

A family $\mathcal{F}'$ a simplification *of $\mathcal{F}$ if it is simple and every formula piecewise in $\mathcal{F}$ is also piecewise in $\mathcal{F}'$.*

▶ **Proposition 23.** *Suppose $\mathcal{F}$ is a simple family and φ, ψ are piecewise in $\mathcal{F}$. Then $\varphi \wedge \psi$ and $\varphi \vee \psi$ are piecewise in $\mathcal{F}$.*

▶ **Proposition 24.** *Every finite family $\mathcal{F}$ of basic formulae has a unique (subset) minimal simplification, which can be computed in polynomial time.*

Towards proving Lemma 9, recall that $Vanish_X(x, y)$ holds if one can rewrite $(X, x) \xrightarrow{y} \emptyset$. We will characterize these predicates using a type of game. These are essentially one-dimensional priced timed games of [13, 28], with the provisio that Maximizer can only make discrete steps and cost rates can only be 0 or 1. Additionally, we are interested in identifying if *optimal* (value achieving) minimizer strategies exists, which is why we are dealing with value *formulae* instead of simply value functions.

▶ **Definition 25.** *A* VGame *is given by a timed automaton $\mathcal{A} = (\mathcal{C}, \mathcal{X}, \mathcal{R})$, a partitioning $\mathcal{X} = \mathcal{X}_{min} \cup \mathcal{X}_{\max}$ of the set of states into those belonging to players Min and Max, respectively, and a* cost $\mathrm{rate}(X) \in \{0, 1\}$ *for every state $X \in \mathcal{X}$. Additionally, a game is parametrized by a payoff predicate* $\mathrm{payoff}_X(x, y)$ *for every state X.*

A play *of the game is a walk $(s_0, t_0), (s_1, t_1), (s_2, t_2), \ldots$ in the configuration graph of $\mathcal{A}$, where the player who owns the current state chooses the successor, with the restriction that only Min can use time steps.*

The outcome *of infinite plays is ∞. At any point in the play where the current configuration $(s_i, t_i) \in \mathcal{X}_{\min} \times \mathbb{R}$ belongs to Min and* $\mathrm{payoff}_{s_i}(t_i, p)$ *holds for some $p \in \mathbb{R}$, she can stop the play and get the finite outcome $p + \sum_{i=0}^{n-1} \mathrm{rate}(s_i) \cdot (t_{i+1} - t_i)$. The aim of players Min/Max is to minimize and maximize the outcome, respectively.*

The outcome formulae *are the family $\{\varphi_X \mid X \in \mathcal{X}\}$ of formulae such that $\varphi_X(x, y)$ holds iff Min has a strategy to guarantee a payoff of at most y from initial configuration (X, x).*

In order to compute the $Vanish_X$ predicates for a given TBPP we can equivalently compute the outcome formulae for a corresponding VGame, where Min gets to pick derivation rules and Max gets to pick the successor whenever a branching rule is used. Every nonterminal owned by Min has cost rate 1, and all other have rate 0. The payoff formula for $X \in \mathcal{X}$ is simply $\text{payoff}_X(x,y) \equiv \bigvee\{\varphi(x) \mid (X \overset{\varphi;R}{\Longrightarrow} \emptyset) \in \mathcal{R}\}$, i.e., it is true iff $(X,x) \xrightarrow{0} \emptyset$, the configuration (X,x) can vanish in no time.

We will solve these games iteratively, based on the special case for a single clock interval.

▶ **Definition 26.** *Let $a,b \in \mathbb{R}$. An (a,b)-game is a VGame $\mathcal{G} = (\mathcal{C},\mathcal{X},\mathcal{R},\text{rate},\text{payoff})$ such that every rule $(X \overset{\varphi;R}{\Longrightarrow} \alpha) \in \mathcal{R}$ has the same clock constraint $\varphi(x) \equiv a < x < b$ and does not reset the clock, and all payoff predicates are basic and with domain (a,b).*

Let $\mathcal{F}$ be the family of formulae that contains for every $\text{payoff}_X(x,y) \equiv (\varphi(x) \wedge y \diamond c)$, the two formulae $(\varphi(x) \wedge y \diamond c)$ and $(\varphi(x) \wedge y \diamond c - (b-x))$. Notice (Proposition 21) that $\mathcal{F}$ contains only basic formulae, and that there are at most $|\mathcal{X}|$ points at which two formulae from $\mathcal{F}$ can cross. We write $\mathcal{F}(\mathcal{G})$ for the simplification of $\mathcal{F}$.

The following properties of $\mathcal{F}(\mathcal{G})$ follow from easy geometric reasoning.

▶ **Proposition 27.**
- *$\mathcal{F}(\mathcal{G})$ has at fewer than $4|\mathcal{X}|^3$ many elements.*
- *Every formula piecewise in $\mathcal{F}(\mathcal{G})$ is a disjunction of at most $2|\mathcal{X}|(|\mathcal{X}|-1)+1$ many basic formulae.*
- *If a formula f is piecewise in $\mathcal{F}(\mathcal{G})$ then so is $\bigvee_{0\le t\le(b-x)}(f(x+t, y-\text{rate}(X)\cdot t)$.*

▶ **Lemma 28.** *The outcome formulae for the (a,b)-game $\mathcal{G}$ are piecewise in $\mathcal{F}(\mathcal{G})$ and computable in polynomial time.*

Proof. First, without loss of generality we may assume that there are no cycles in the game restricted only to the Max player states. If they are then their outcome must be $\text{outcome}_X(x,y) \equiv \textbf{false}$ as Max can enforce an infinite play. Further, we assume that every transition going from a Max configuration is going to a Min player configuration, we simply introduce shortcuts for Max player.

Using the natural partial ordering $f \le g$ iff $f(x,y) \implies g(x,y)$ of two-variable formulae, we notice that the outcome formulas $\{g_X \mid X \in \mathcal{X}\}$ must be the least family of formulae satisfy the following equations.

$$g_X(x,y) \quad = \quad \bigwedge_{X \overset{\varphi}{\Longrightarrow} Y} g_Y(x,y) \qquad \text{for } X \in \mathcal{X}_{\max}, \text{ and} \tag{2}$$

$$\begin{aligned} g_X(x,y) \quad = \quad \text{payoff}_X(x,y) \quad &\vee \bigvee_{(X \overset{\varphi}{\Longrightarrow} Y) \in \mathcal{R}} g_Y(x,y) \\ &\vee \bigvee_{0<t\le(b-x)} g_X(x+t, y-\text{rate}(X)\cdot t) \end{aligned} \tag{3}$$

for $X \in \mathcal{X}_{min}$. This allows to approximate them as the least fixed point of ordinal indexed families $\mathcal{F}^{(\alpha)}$, where $\mathcal{F}^{(0)} \overset{def}{=} \{g_X^{(0)} \overset{def}{=} \text{payoff}_X \mid X \in \mathcal{X}_{\min}\} \cup \{g_X^{(0)} \overset{def}{=} \textbf{false} \mid X \in \mathcal{X}_{\max}\}$, and for all $\alpha > 0$ define

$$g_X^{(\alpha)}(x,y) \quad \overset{def}{=} \quad \bigwedge_{X \overset{\varphi}{\Longrightarrow} Y} g_Y^{(\alpha)}(x,y) \qquad \text{for } X \in \mathcal{X}_{max}, \text{ and} \tag{4}$$

$$\bigvee_{\beta<\alpha}\left(g_X^{(\beta)}(x,y) \quad \vee \bigvee_{X\overset{\varphi}{\Longrightarrow}Y} g_Y^{(\beta)}(x,y) \quad \vee \bigvee_{0<t\leq(b-x)} (g_X^{(\beta)}(x+t, y-\mathrm{rate}(X)\cdot t))\right) \tag{5}$$

for $X \in \mathcal{X}_{\min}$. Notice that here, $g_Z^{(\alpha)} \leq g_X^{(\beta)}$ for $\alpha < \beta$. Intuitively, $g_X^{(\alpha)}(x,y)$ holds if in a game starting in (X,x), Min only needs to make α moves to get a payoff of at least y.

▷ Claim 29. Every family $\mathcal{F}^{(k)}$ with finite index $k \in \mathbb{N}$ is piecewise in $\mathcal{F}(\mathcal{G})$.

We prove this by induction on k, where the base case, $\mathcal{F}^{(0)} =$ payoff trivially holds. For the induction step, consider the case for $X \in \mathcal{X}_{\min}$ and let

$$g_X^{(k+1)}(x,y) \quad = \quad g_X^{(k)}(x,y) \vee \left(\bigvee_{X\overset{\varphi}{\Longrightarrow}Y} g_Y^{(k)}(x,y) \quad \vee \bigvee_{0<t\leq(b-x)} (g_X^{(k)}(x+t, y-\mathrm{rate}(X)\cdot t))\right) \tag{6}$$

Here, $(g_X^{(k)}(x+t, y-\mathrm{rate}(X)\cdot t))$ is piecewise in $\mathcal{F}(\mathcal{G})$ by (the last point in) Proposition 27, because $g_X^{(k)}$ is piecewise in $\mathcal{F}(\mathcal{G})$ by induction assumption. Together with Proposition 23 this shows that $g_X^{(k+1)}$ must be piecewise in $\mathcal{F}(\mathcal{G})$. The case for Maximizer states is analogous.

▷ Claim 30. There exists $k \leq 4|\mathcal{X}|^3$ such that $\mathcal{F}^{(k+1)} = \mathcal{F}^{(k)}$.

This follows from Claim 29 and Proposition 27: Every time $\mathcal{F}^{(k)}$ strictly improves there must be at least one of its elements $g_X^{(k)}(x,y) = \bigvee_{i\leq|\mathcal{X}|} f_i$ where at least one disjunct f_i is replaced by a strictly larger one. This can happen at most $4|\mathcal{X}|^3$ times.

It remains to observe that one can compute a representation of $\mathcal{F}^{(k+1)}$ from $\mathcal{F}^{(k)}$, using Equations (4) and (5). ◀

Lemma 9 now rests on the following lemma, which we prove by recursively applying Lemma 28 to solve VGames without resets and transition guards.

▶ **Lemma 31.** *Let* $(\mathcal{C}, \mathcal{X}, \mathcal{R}, \mathrm{rate}, \mathrm{payoff})$ *be a VGame where* payoff *is piecewise basic. Let* $k \in \mathbb{N}$ *denote the number of different constants appearing either as endpoints in* payoff *or in the guards of rules. Every outcome formula* $\mathrm{outcome}_X(x,y)$ *is a disjunction of at most* $2 \cdot k \cdot |\mathcal{X}|$ *basic formulae and is computable in polynomial time.*

Proof. We can partition the reals into $2k+1$ intervals according to the constraint constants

$$\{a_0 = 0\}, (a_0, a_1), \{a_1\}, (a_1, a_2)\{a_2\}, \ldots (a_{k-1}, a_k = \infty)$$

and write $\mathrm{outcome}_X(x,y) \equiv \bigvee_{i\leq k} f_X^i$, where $f_X^{2i}(x,y) \overset{def}{=} \mathrm{outcome}_X(x,y) \wedge (x = a_i)$ for even indices $f_X^{2i+1}(x,y) \overset{def}{=} \mathrm{outcome}_X(x,y) \wedge (a_i < x < a_{i+1})$ for odd indices.

Suppose first that no rule in the VGame resets a clock to 0. In this case f_X^i are independent of f_X^j with smaller index $j < i$ and can be computed stepwise from last to first interval, preserving the invariant that each f_X^i is the disjunction of at most $|\mathcal{X}|$ many basic formulae.

For the final interval we must have $f_X^{2k+1}(x,y) \equiv (x > a_k) \wedge \mathrm{payoff}_X(x,y)$.

- For even indices (singleton intervals $\{a_i\}$), outcome formulae can be computed once $\{f_Y^{2i+1} \mid Y \in \mathcal{X}\}$ are known, using dynamic programming. Briefly, compute payoff formulae of the form $\phi_X(x,y) \equiv (x = a_i) \wedge y \diamond c$, for $\diamond \in \{>, \geq\}$ and $c \in \mathbb{R}$, from the f^{2i+1}, which is possible because those are piecewise basic and by Proposition 21, and solve an untimed min/max game for those payoffs.
- For odd indices corresponding to intervals (a_i, a_{i+1}), we can use Lemma 28 to compute f_X^{2i+1} as the outcome formulae for an (a_i, a_{i+1})-game with payoffs given by $\text{payoff}_X(x,y) \equiv (x = a_{i+1}) \wedge f_X^i(x,y)$.

This produces outcome formulae which are piecewise basic and expressed as disjunctions of at most $2k|\mathcal{X}| + 1$ basic formulae.

It remains to argue that the above construction can be extended to VGames *with* resetting rules. This is based on the observation that Minimizer can be required not to allow a configuration to repeat along a play. Indeed, the outcome of finite a play

$$(s_0, t_0), \ldots, (s_0, t_0), \ldots, (s_k, t_k)p$$

where $p \in \mathbb{R}$ as chosen by Min at the end of a finite play, can only be larger than that of the suffix $(s_0, t_0), \ldots, (s_k, t_k)p$ alone. Any Min strategy that allows for repetitions can therefore be turned into one that does not, by cutting out the intermediate paths. Consequently, plays with more than $|\mathcal{X}|$ clock resets can be declared to be losing (have outcome ∞) without changing the resulting outcome formulae.

In order to compute outcome formulae for an unrestricted VGame we can now stepwise compute formulae $\text{outcome}_X^{(i)}$ for the same game but where a play ends (with outcome ∞) as soon as Min uses the ith reset. By the observation above, $\text{outcome} = \text{outcome}^{|\mathcal{X}|}$. The base case $\text{outcome}_X^{(0)}$ corresponds to the game where all reset rules are removed.

In order to compute $\text{outcome}_X^{(i+1)}$ we can use the same procedure as described above, but in every (a,b)-game, we replace resetting rules $X \xRightarrow{\varphi, x:=0} Y$ by non-resetting rules $X \xRightarrow{\varphi} Y'$ to new, unproductive nonterminals $\hat{Y}$ with $\text{payoff}_{\hat{Y}}(x,y) \stackrel{def}{=} (x = b) \wedge \text{outcome}_Y^{(i)}(0,y)$, and $\text{rate}(\hat{Y}) = 0$. This way, if from (X,t), Min would originally gain an outcome by resetting to $(Y,0)$, she can now gain the same outcome by moving to (Y',t), delaying by $(b-t)$ at no cost, and collect the outcome in configuration $(\hat{Y},b)$. This change will not introduce any new constants in the guards of rules, so the total number of intervals in the description of the outcome formulae remains unchanged. ◀